The Role of Conceptual Structure in Designing Cellular Automata to Perform Collective Computation

Manuel Marques-Pita[1,2,3], Melanie Mitchell[3], and Luis M. Rocha[1,2]

[1] Indiana University
[2] Instituto Gulbenkian de Ciência
[3] Portland State University

Abstract. The notion of conceptual structure in CA rules that perform the density classification task (DCT) was introduced by [1]. Here we investigate the role of *process-symmetry* in CAs that solve the DCT, in particular the idea of *conceptual similarity*, which defines a novel search space for CA rules. We report on two new process-symmetric one-dimensional rules for the DCT which have the highest "balanced" performance observed to date on this task, as well as the highest-performing CA known to perform the DCT in two dimensions. Finally, we investigate the more general problem of assessing how different learning strategies (based on evolution and coevolution, with and without spatial distribution), previously compared by [2], are suited to exploit conceptual structure in learning CAs to perform collective computation.

1 Introduction

The study of computation in cellular automata (CAs) and related cellular architectures has lately garnered renewed interest due to advances in the related fields of reconfigurable hardware, sensor networks, and molecular-scale computing systems. In particular, cellular array architectures are thought to be appropriate for constructing physical devices such as field configurable gate arrays for electronics, networks of robots for environmental sensing and nano-devices embedded in interconnect fabric used for fault tolerant nanoscale computing [3]. A current stumbling block for CA computing is the difficulty of programming CAs to perform desired computations, due to the decentralized architectures and nonlinear behavior of these systems. One approach is to use genetic algorithms or other evolutionary computation methods to evolve cellular automata transition rules that will perform desired computations. However, this approach has problems of scaling, due to the large search spaces for non-elementary CAs—those with larger than nearest-neighbor cell communication or with multiple states per cell.

In this paper we describe our investigation of reducing the dimensionality of these search spaces by using automatically-discovered *conceptual structures* of rule tables that are common to CAs likely to be successful for a particular computational task. We show that for one well-studied task—two-state density classification—a particular conceptual structure of CA rule tables that we call *degree of process symmetry* is correlated with success on the task, and is implicitly increased by genetic algorithms evolving CAs for this task. We also show that process symmetry provides a search space of significantly reduced dimensionality, in which a genetic algorithm can more easily discover high-performing one- and two-dimensional CAs for this task.

2 Cellular Automata

A cellular automaton (CA) consists of a regular lattice of N cells. Each cell is in one of k allowed states at a given time t. Let $\omega \in \{0, 1, ..., k-1\}$ denote a possible state of a cell. Let state $\omega = 0$ be referred to as the *quiescent* state, and any other state as an *active* state. Each cell is connected to a number of neighbors. Let a local neighborhood configuration (LNC) be denoted by μ, and its size by n. For each LNC in a CA an output state is assigned to each cell. This defines a CA rule string, ϕ, the size of which is k^n. In binary CAs, in which only two states are allowed($k = 2$), it is possible to classify individual cell state-updates in three categories: (1) *preservations,* where a cell does not change its state in the next time instance $t+1$; (2) *generations,* state-updates in which the cell goes from the quiescent to the active state; and (3) *annihilations,* state-updates where the cell goes from the active to the quiescent state. The execution of a CA for a number M of discrete time steps, starting with a given initial configuration (IC) of states, is represented as the set Θ containing $M+1$ lattice state configurations.

2.1 The Density Classification Task

The Density Classification Task (DCT) is a widely cited example of collective computation in cellular automata. The goal is to find a one-dimensional binary CA rule (with periodic boundary conditions) that can classify the majority state in a given, random IC (with odd number of cells). If the majority of cells in the IC are in the quiescent state, after a number of time steps M, the lattice should converge to a homogeneous state where every cell is in the quiescent state, with analogous behavior for an IC with a majority of active cells. Devising CA rules that perform this task is not trivial, because cells in a CA lattice update their states based only on local neighborhood information. However, in this particular task, it is required that information be transferred across time and space in order to achieve a correct global classification. The definition of the DCT used in our studies is the same as the one given by [4].

We define the *performance* $\mathcal{P}_N^K(\phi)$ the fraction of K initial configurations on a N-cell lattice that produce correct classifications (all quiescent for a majority of quiescent states in the IC; all active for a majority of active states in the IC).

Nine of the cellular automata rules with highest performance on the DCT were analyzed to determine whether there is *conceptual structure* not explicit in them, and if so, to investigate the possible *conceptual similarity* among them using a cognitively inspired mechanism (Aitana) [1]. Three of these rules were produced by human engineering: ϕ_{GKL} [5, 6], $\phi_{Davis95}$ and ϕ_{Das95} [7]; three were learned with genetic algorithms ϕ_{DMC} [8] or coevolution methods ϕ_{COE1} and ϕ_{COE2} [9]. Finally, three of the rules were learned with genetic programming or gene expression programming: ϕ_{GP1995} [7], ϕ_{GEP1} and ϕ_{GEP2} [10]. The next section summarizes the basics of Aitana's architecture, and the conceptual properties found in the studied CAs that perform the DCT.

3 CA Schemata Redescription

Aitana is largely based on an explanatory framework for cognitive development in humans known as the *Representational Redescription Model* developed by [11], and the *Conceptual Spaces* framework proposed by [12]. There are a number of (recurrent) phases in Aitana's algorithm: (1) *Behavioral Mastery,* during which CAs that perform some specific collective computation are learned using, for example, genetic algorithms or coevolution. The learned rules are assumed to be in a representational format we call *implicit* (conceptual structure is not explicit). (2) *Representational Redescription Phase I* takes as input the implicit representations (CA look-up tables) and attempts to compress them into *explicit-1* (E1) schemata by exploiting structure within the input rules. (3) *Phase II* and beyond look for ways to further compress E1 representations, for example by looking at how groups of cells change together, and how more complex schemata are capable of generating regular patterns in the dynamics of the CA. The focus in this paper is on Phase I redescription.

E1 representations in Aitana are produced by different *modules.* In particular, two modules were explored by [13]: the *density* and *wildcard* modules. Modules in Aitana can be equated to representational transducers, where each module takes implicit CA rules, and outputs a set of E1 schemata that redescribe them. The nine high-performing CA rules we report on here were redescribed with the wildcard module, introduced in the next section.

3.1 The Wildcard Module

This module uses regularities in the set of entries in a CA's look-up table, in order to produce E1 representations captured by *wildcard schemata.* These schemata are defined in the same way as the look-up table entries for each LNC of a CA rule, but allowing an extra symbol to replace the state of one or more cells within them. This new symbol is denoted by "#". When it appears in a E1 schema it means that in the place where it appears, any of the possible k states is accepted for state update. The idea of using wildcards in representational structures was first proposed by [14], when introducing Classifier Systems. Wildcard schemata can be *general* or *process-specific.* The first variation allows wildcards to appear in the position of the updating cell in any schema. Process-specific schemata do not allow this, therefore making it possible for them to describe processes in the CA rule unambiguously. For example, given a one-dimensional, binary CA with local neighborhoods of length 7, a *generation*, *process-specific*, wildcard schema $\{\#, \#, \#, 0, 1, \#, 1\} \rightarrow 1$ prescribes that a cell in state 0, with immediate-right and end-right neighbors in state 1 updates its state to 1 regardless of the state of the other neighbors. The implementation of the wildcard module in Aitana consists of a simple McCulloch and Pitts neural network that is instantiated distinctly for each combination of values for neighborhood size n, and number of allowed states k of an input CA rule. In this *assimilation network,* input units represent each look-up table entry (one for each LNC), and ouput units represent all the schemata available to redescribe segments of the input rule (see [13]).

3.2 Assimilation and Accommodation

Phase I redescription in Aitana depends on two interrelated mechanisms, *assimilation* and *accommodation*[4]. During Phase I, the units in the input layer of an assimilation network will be activated according to the output states in the CA rule to be processed. The firing of these units will spread, thus activating other units across the network. When some unit in the network (representing a E1 schema) has excitatory input above a threshold it fires. This firing signals that the schema represented by the unit becomes an E1 redescription of the lower level units that caused its activation. When this happens, inhibitory signals are sent back to those lower level units so that they stop firing (since they have been redescribed). At the end of assimilation, the units that remain firing represent the set of wildcard schemata redescribing the input CA rule. Once the process of assimilation has been completed, Aitana will try to force the assimilation of any (wildcard-free) look-up table entry that was not redescribed *i.e.* any input unit that is still firing. This corresponds to the accommodation process implemented in Aitana [13].

4 Conceptual Structure

One of the main findings reported in [1] is that most rules that perform the density classification task are *process-symmetric*. A binary CA rule is defined as process-symmetric if a particular bijective mapping (defined below) maps each schema representing a *generation* into exactly one of the schemata representing an *annihilation*, and vice versa.

The bijective function transforms a schema s into its corresponding process-symmetric schema s' by (1) reversing the elements in s using a *mirror* function $M(s)$, and (2) exchanging ones for zeros, and zeros for ones (leaving wildcards untouched), using a *negation* function $N(s)$. Thus, in every process symmetric CA rule, given the set $S = \{s_1, s_2, ..., s_z\}$ of all schemata s_i prescribing a state-change process, the elements of the set of schemata prescribing the converse process $S' = \{s'_1, s'_2, ..., s'_z\}$ can be found by applying the bijective mapping between processes defined by the composition $s'_i = (M \circ N)(s_i)$.

[4] These two processes are inspired in those defined by Piaget in his theory of Constructivism [15, 16]

Six out the nine rules analyzed by [1] were found to be process-symmetric. The remaining three, ϕ_{COE1} and ϕ_{COE2} and ϕ_{DMC} are not. It is interesting to note that the latter three CA rules were discovered via evolutionary algorithms (GAs and coevolutionary search) which apply variation to genetic encodings of the look-up tables of CAs. Therefore, genotype variation in these evolutionary algorithms operates at the low level of the bits of the look-up table—what we referred to as the implicit representation of a CA. In contrast, the search (Genetic Programming and Gene Expression Programming) and human design processes that lead to the other six (process-symmetric) rules, while not looking explicitly for process symmetry, were based on mechanisms and reasoning trading in the higher-level behavior and structure of the CA—what we refer to as the explicit representation of a CA[5] The same research also determined that it is possible to define conceptual similarity between the process symmetric CA rules for the DCT. For example, the rule ϕ_{GP1995} can be derived from ϕ_{GKL} [1]. Moreover, the best process-symmetric rule known for this task (at the time) was found via conceptual transformations: ϕ_{MM401}[6] with performance $\mathcal{P}_{149}^{10^5} \approx 0.83$. However, this still below the performance of the highest-performance rule so far discovered for the DCT, namely ϕ_{COE2}, with $\mathcal{P}_{149}^{10^5} \approx 0.86$.

5 The 4-Wildcard Space

Starting with the conceptual similarities observed between ϕ_{GKL} and ϕ_{GP1995}, we studied the "conceptual space" in which these two CA rules can be found: the space of process-symmetric binary CA rules with neighborhood size $n = 7$, where all state-change schemata have four wildcards. A form of evolutionary search was used to evaluate rules in this space as follows: the search starts with a **population** of sixty-four different process-symmetric rules containing only 4-wildcard schemata; the generation and annihilation **schema sets** for an individual were allowed to have any number of schemata in the range between two and eight; **crossover operators** were not defined; a **mutation operator** was set, allowing the removal or addition of up to two randomly chosen 4-wildcard schemata (repetitions not allowed), as long as a minimum of two schemata are kept in each schema set; in every **generation** the fitness of each member of the population is evaluated against 10^4 ICs, keeping the top 25% rules (elite) for the next generation without modification; **offspring** are generated by choosing a random member of the elite, and applying the mutation operator until completing the population size with different CA rules; **a run** consisted of 500 generations, and the search was executed for 8 runs.

[5] When we refer to implicit and explicit representations of CA rules, we are preserving the terminology of the Representational Redescription Model (§3), the basis of the cognitively-inspired Aitana. We do not mean to imply that state-transition rules of CA are implicit, but rather that it is not clear from these rules what conceptual properties they embody.

[6] In inverse lexicographical (hex) , ϕ_{MM401} is `ffaaffa8ffaaffa8f0aa00a800aa00a8`

There are 60 possible 4-wildcard process-symmetric schemata-pairs. Thus, our search space contains approximately 3×10^9 rules defined by generation and annihilation schema sets of size between 2 and 8. As reported in [17], our search found one rule with higher performance than ϕ_{MM401}. This rule, ϕ_{MM0711}[7] has $\mathcal{P}_{149}^{10^5} \approx 0.8428$. Even though this search resulted in an improvement, the performance gap between the best process-symmetric rule, ϕ_{MM0711} and ϕ_{COE2} is still close to 2%. Is it possible then, that a process-symmetric rule exists "hidden" in the conceptually "messy" ϕ_{COE2}?

6 Process-Symmetry in ϕ_{COE2}

Figure 1 shows the state-change schema sets for ϕ_{COE2}. The performance of this rule is $P_{149}^{10^5} \approx 0.86$. We generated random ICs (binomial distribution with $\rho = 0.5$), where each IC was put in one of two sets—with membership to each depending on whether the IC has majority 0's or 1's. This was done until each set contained 10^5 ICs. Then the DCT performance measure was calculated for the sets of ICs. These were, respectively, $P_{149}^{10^5}$(majority-0 ICs) ≈ 0.83 and $P_{149}^{10^5}$(majority-1 ICs) ≈ 0.89. Even though on average this is the best-performing CA rule for the DCT, its performance is noticeably higher on the majority-1s set of ICs. We claim that this divergence in behavior is due to the fact that ϕ_{COE2} is not process-symmetric. Evaluation of split performance on the ten known highest-performing rules for the DCT supports this hypothesis (see Table 1). The difference between the split performance measures for the non-process-symmetric rules is one or two orders of magnitude larger than for the process-symmetric rules. This indicates that process symmetry seems to lead to more balanced rules—those that respond equally well to both types of of problem.

RULE	Generation	Annihilation

ϕ_{COE2}	g1 {1, 0, 1, 0, #, #, #}
	g2 {1, 0, #, 0, #, 1, 1}
	g3 {1, 1, #, 0, 1, #, #}
	g4 {1, #, 1, 0, 1, #, #}
	g5 {1, #, 1, 0, #, 0, #}
	g6 {1, #, #, 0, 1, 1, #}
	g7 {1, #, #, 0, 1, #, 1}
	g8 {#, 0, 0, 0, 1, 0, 1}
	g9 {#, 0, 1, 0, 0, 1, #}
	g10 {#, 0, #, 0, 0, 1, 1}
	g11 {#, 1, 1, 0, 1, #, 0}
	g12 {#, 1, 1, 0, #, 0, #}
		a1 {0, 0, 1, 1, 1, 1, #}
		a2 {0, 0, #, 1, #, 1, 0}
		a3 {0, 1, 0, 1, 1, #, #}
		a4 {0, #, 0, 1, #, #, 0}
		a5 {1, 0, 0, 1, #, 0, #}
		a6 {#, 0, 0, 1, #, #, 0}
		a7 {#, #, 0, 1, 1, 0, #}
		a8 {#, #, 0, 1, #, 0, 0}
		a9 {#, #, #, 1, 0, #, 0}

Fig. 1. E1 schemata prescribing state changes for ϕ_{COE2}. This is the highest performance rule for the DCT found to date, and does not show clear process symmetry.

[7] ϕ_{MM0711} is `faffba88faffbaf8fa00ba880a000a88`

	$P_{149}^{10^5}$ M→0	$P_{149}^{10^5}$ M→1	*P*. DIFF.
ϕ_{GKL}	0.8135	0.8143	0.0008
$\phi_{Davis95}$	0.8170	0.8183	0.0013
ϕ_{Das95}	0.8214	0.8210	0.0004
ϕ_{GP1995}	0.8223	0.8245	0.0022
ϕ_{DMC}	0.8439	0.7024	0.1415
ϕ_{COE1}	0.8283	0.8742	0.0459
ϕ_{COE2}	0.8337	0.888	0.0543
ϕ_{GEP1}	0.8162	0.8173	0.0011
ϕ_{GEP2}	0.8201	0.8242	0.0041
ϕ_{MM0711}	0.8428	0.8429	0.0001

Table 1. Split performances of the ten best DCT rules. First column shows performance for ICs in which there is majority of 0s; the is the performance when ICs have majority 1s; the third shows the difference between the two performances. Darker rows correspond to process-symmetric rules; white rows refer to non-process-symmetric rules.

A relevant question at this point concerns the existence of a process-symmetric rule in the conceptual vicinity of ϕ_{COE2}, whose performance is as good (or higher) than the performance of the original ϕ_{COE2}. There are two ways in which it is possible to think about conceptual vicinities, where new neighboring rules are produced by different accommodation mechanisms. One approach is to work with schemata that are in the original set describing the analyzed rule only. In this context, it is possible to produce new rules by deleting schemata (e.g. deleting a schema from a generation set, the process symmetric of which is not in the original annihilation set), or by adding process symmetric schemata to a set, provided their process symmetric counterparts are present in the original rule. We will refer to this as the *"naive" approach to accommodation.* Note that accommodation here has the goal of generating process-symmetric rules, instead of ensuring full assimilation as described in §3.2. A second approach would be to work with manipulations on the LNC (implicit) representational level – with these followed by a necessary re-assimilation of the manipulated rule. This type of accommodation will produce new sets of schemata that replace (fully or partially) the ones in the original rule, due to the fact that the LNCs in the rule were manipulated. This approach will be referred to as the *"re-conceptualization" approach to accommodation.*

When working with rules such as ϕ_{COE2}, which were evolved by learning mechanisms that are unaware of process symmetry, the first approach just described is "naive". It is so in the sense that it is likely that evolution produced pairs of schemata (for generation and annihilation) which are only partially process-symmetric; they may contain key process-symmetric LNC pairs that are necessary to perform the computation.Thus, simply adding and deleting schemata to attain process-symmetry may miss subtle interactions present on the implicit representational level of a CA rule. This makes the naive approach too "coarse" for dealing with CA rules evolved with learning strategies that do not take process symmetry into account.

In answer the question about of the possible existence of a process-symmetric rule in the conceptual vicinity of ϕ_{COE2}, we performed a number of tests. First, using the naive approach, we looked at the CA rule resulting from keeping all annihilations in ϕ_{COE2}, and using only their process-symmetric generations. The performance of that rule was $P_{149}^{10^5} \approx 0.73$. A second test was the reverse of the first: keeping all generations of ϕ_{COE2}, and using only their process-symmetric annihilations. The resulting rule has a performance $P_{149}^{10^5} \approx 0.47$—a large decrease in performance.

In order to interpret the results of these first two tests it would be necessary to study how different schemata and LNCs interact when they form coherent time-space patterns. The set of annihilations in ϕ_{COE2} seems to contribute more to the overall collective computation than the set of original generations, since this set of annihilation schemata by itself, working with its corresponding process symmetric generation set, results in a CA with significant higher performance than for the other case (second test). Nonetheless, the naive (coarser) approach to accommodation did not "uncover" a process symmetric rule in ϕ_{COE2} that keeps (or improves) the original average performance.

For the next test, we used the "finer" approach to accommodation in order to explore the conceptual vicinity of ϕ_{COE2} plus some additional constraints (explained later). First of all, we looked at the *degree of process symmetry* already existing in ϕ_{COE2}. To find this we used the matrix-form representation of ϕ_{COE} illustrated in Figure 2. Each column corresponds to each of the 128 LNCs for a one-dimensional binary CA rule and neighborhood radius three. These LNCs are not arranged in lexicographical order, instead they are arranged as process-symmetric pairs: the first and last LNCs (columns) are process-symmetric, the second, and next to last are also process-symmetric and so on, until the two LNCs in the center are also process-symmetric. Each row corresponds to the E1 (wildcard) state-changing schemata for ϕ_{COE2}. The first nine rows correspond to the annihilation schemata, and the subsequent ones the twelve generation schemata for ϕ_{COE2}. In any of the first nine rows, a shaded-cell represents two things: (1) that the LNC in that column is an annihilation; and (2) that the LNC is part of the E1 schema labeled in the row where it appears. The twelve rows for generation schemata are reversed in the figure. This makes it simple to inspect visually what process-symmetric LNCs are present in the rule, which is the case when for a given column, there is, at least, one cell shaded in one of the first nine rows (an active annihilation, light gray), and at least one cell shaded in one of the bottom nine rows (an active generation, dark gray). We will refer the *schemata x LNC* matrix representation in Figure 2 as A.

As just described, given the ordering of elements in the columns of Figure 2, if a generation row is isolated, and then reversed, the result can be matched against any of the annihilation rows to calculate the total degree of process symmetry between the two schemata represented in the two rows. A total match means that the original generation schema is process-symmetric with the matched annihilation schema. A partial match indicates a degree of process symmetry. This partial match can be used by Aitana's accommodation mechanism to force the highly process-symmetric pair into a fully process-symmetric one, keeping the modified representation only if there is no loss of performance.

More concretely, the degree of process symmetry existing between two schemata S_g and S_a prescribing opposite processes (a generation schema, and an annihilation respectively) is calculated as follows:

1. Pick rows S_g and S_a from matrix A; S_g corresponds to a generation and S_a to an annihilation).
2. Reverse one of the rows (e.g. S_a. This makes it possible to compare each LNC (the columns) with its process-symmetric pair, by looking at the i^{th} element of each of the two row vectors.
3. Calculate the degree of process symmetry as:

   $$\frac{2 \times S_g \cdot S_a}{|S_g| + |S_a|}$$

   where, for binary vectors, $S_g \cdot S_a$ is the number of component-matches (i.e. the count of all of the i^{th} components that are one in both vectors); and $|S|$ is the number of ones in a binary vector.[8]

All the generation rows were matched against all the annihilation rows in matrix A, recording the proportion of matches found. Table 3 (A) shows the results of this matching procedure (only highest matches shown). The darker rows correspond to schema pairs that are fully process-symmetric. The first three light gray rows (with matching score 66% show an interesting, almost complete process symmetry subset, involving generation schemata *g1, g4* and *g5,* and annihilation schema *a9*.

Using the accommodation mechanism in Aitana, we "generalized" schemata *g1, g4* and *g5* into the more general process symmetric schema of *a9* (that encompasses the three generation processes), and tested the resulting CA rule. We also "specialized" by breaking *a9* into the three process-symmetric schemata of *g1*, *g4* and *g5*, and forcing the remaining LNCs to become preservations. For both the resulting rules performance decreased significantly, $P_{149}^{10^5} < 0.6$. Notice that for these tests, the approach used to define what rules are in the conceptual vicinity is more fined-grained, but still constrained to working with schemata, allowing mechanisms such as the generalization of schemata e.g. *g1*, *g4* and *g5* into a single one to work. However, these tests were also unsuccessful in uncovering a high-performing CA derived from ϕ_{COE2}.

Using the re-conceptualization approach, it is possible to extract a matrix representation A' that contains *only* those LNC process-symmetric pairs that

[8] While $|x|$ is the notation typically used for cardinality of sets, here, we use it to represent the 1-norm, more commonly denoted by $||x||_1$.

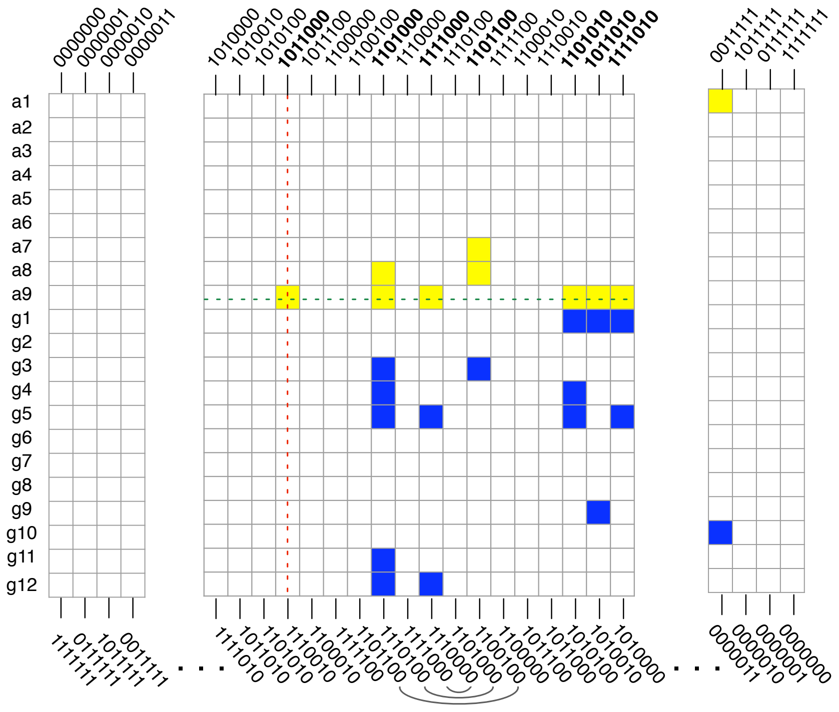


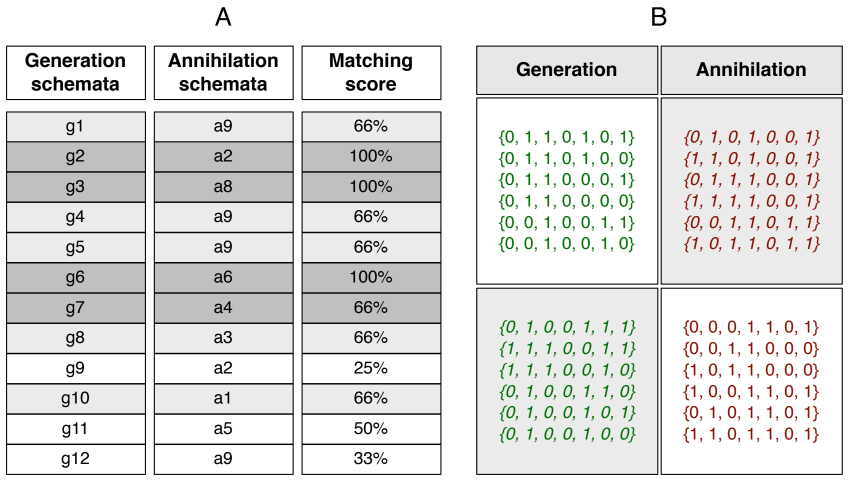
Fig. 2. Representation of the matrix A used to determine the degree of processes symmetry for a CA rule (here ϕ_{COE2}). Of the 128 possible LNCs only the first and last four, plus the middle eighteen are shown. Matrix elements colored in the first nine rows correspond to annihilation LNCs (labeled at the top). Analogously, the darker elements in the bottom twelve correspond to generation LNCs (labelled at the bottom). The curved connecting lines represent the ordering of the columns as process symmetric pairs; the vertical dotted line represents an annhilation LNC that is not process symmetric; and the horizontal dotted line represents that annihilation LNCs in that row are part of schema *a9*.

are both 1s in A. In other words, each column in A' will be exactly as in A, as long as the column contains 1s for both annihilation and generation rows, otherwise the column is all 0s—the latter is the case for all columns marked with dotted lines in Figure 2. We will refer to the rule represented by the matrix A' as $\phi_{COE2-clean}$—the CA rule that preserves all the process symmetry already present in ϕ_{COE2}. The "orphan" LNCs removed from A are shown in Figure 3 (B) (white background). Their process-symmetric pairs are in the same Figure (gray background). We will refer to this set of process symmetric pairs as R.

The last test to be reported here consisted in evaluating the performance of each CA rule derived from (1) taking $\phi_{COE2-clean}$ as base (each time); (2)

adding to it a number of process symmetric *pairs* from R to it; and (3) evaluating the resulting CA rule. This set contains all CA rules that are the same as $\phi_{COE2-clean}$, but adding *one* of the twelve pairs in R; it also contains all the rules that are as $\phi_{COE2-clean}$, including combinations of two pairs from R (66 rules), and so on. The total number of CA rules derived in this way is 4096[9].

The performance of the 4096 rules is shown in Figure 4 (A). Each column shows the performance of the subsets of rules adding one pair of LNCs from R, subsets adding combinations of two pairs, and so on. Note that the median performance in each subset decreases for rules containing more pairs of LNCs from R. However, the performance of the best CA rules in each subset increases for all subsets including up to six LNC pairs, and then decrease. One of the tested CAs, containing six LNC pairs added to $\phi_{COE2-clean}$, is the best process-symmetric CA for the DCT with $P_{149}^{10^5} \approx 0.85$ ϕ_{MM0802}, are shown in Figure 4 (B). ϕ_{MM0802}, has a performance that is very close to that of the second highest-performing rule known for the DCT, ϕ_{COE1} [1]. However, ϕ_{MM0802} is the highest-performing CA for split performance for the DCT—which means that it classifies correctly the two types of IC it can encounter (majority 1s or majority 0s).



Generation schemata	Annihilation schemata	Matching score
g1	a9	66%
g2	a2	100%
g3	a8	100%
g4	a9	66%
g5	a9	66%
g6	a6	100%
g7	a4	66%
g8	a3	66%
g9	a2	25%
g10	a1	66%
g11	a5	50%
g12	a9	33%

Generation	Annihilation

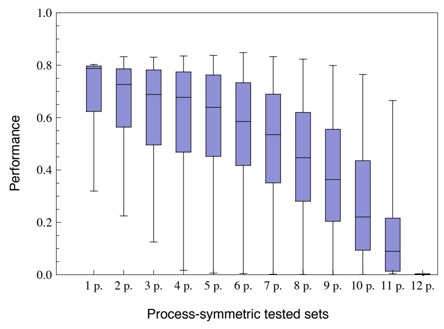
{0, 1, 1, 0, 1, 0, 1}
{0, 1, 1, 0, 1, 0, 0}
{0, 1, 1, 0, 0, 0, 1}
{0, 1, 1, 0, 0, 0, 0}
{0, 0, 1, 0, 0, 1, 1}
{0, 0, 1, 0, 0, 1, 0}
	{0, 1, 0, 1, 0, 0, 1}
	{1, 1, 0, 1, 0, 0, 1}
	{0, 1, 1, 1, 0, 0, 1}
	{1, 1, 1, 1, 0, 0, 1}
	{0, 0, 1, 1, 0, 1, 1}
	{1, 0, 1, 1, 0, 1, 1}

{0, 1, 0, 0, 1, 1, 1}
{1, 1, 1, 0, 0, 1, 1}
{1, 1, 1, 0, 0, 1, 0}
{0, 1, 0, 0, 1, 1, 0}
{0, 1, 0, 0, 1, 0, 1}
{0, 1, 0, 0, 1, 0, 0}
	{0, 0, 0, 1, 1, 0, 1}
	{0, 0, 1, 1, 0, 0, 0}
	{1, 0, 1, 1, 0, 0, 0}
	{1, 0, 0, 1, 1, 0, 1}
	{0, 1, 0, 1, 1, 0, 1}
	{1, 1, 0, 1, 1, 0, 1}

Fig. 3. (A) Degree of process symmetry amongst all the generation and annihilation schemata in ϕ_{COE2}. Darker rows indicate full process symmetry, while light gray rows indicate a high degree of process symmetry. **(B)** The set R, containing the twelve LNCs in ϕ_{COE2} (white background) for which their corresponding process-symmetric LNCs are preservations (gray background).

[9] Note that each of the rules tested comes from adding a particular combination of pairs each time to the original $\phi_{COE2-clean}$, as opposed to adding pairs of LNCs cumulatively to $\phi_{COE2-clean}$.

A



B

RULE	Generation	Annihilation
ϕ_{MM0802}	{1, 0, 1, 0, #, #, #}	{0, 0, 1, 1, 1, 1, #}
	{1, 0, #, 0, #, 1, 1}	{0, 0, #, 1, #, 1, 0}
	{1, 1, #, 0, 1, #, #}	{0, 1, 0, 1, 1, #, #}
	{1, #, 1, 0, 1, #, #}	{0, #, 0, 1, #, #, 0}
	{1, #, 1, 0, #, 0, #}	{1, #, 0, 1, #, 0, #}
	{1, #, #, 0, 1, 1, #}	{#, 0, 0, 1, #, #, 0}
	{1, #, #, 0, 1, #, 1}	{#, 1, 0, 1, #, 0, #}
	{#, 0, 0, 0, 0, 1, 1}	{#, 1, #, 1, 0, #, 0}
	{#, 1, 0, 0, 1, #, #}	{#, #, 0, 1, 0, #, 0}
	{#, 1, #, 0, 1, 0, #}	{#, #, 0, 1, 1, 0, #}
	{#, 1, #, 0, 1, #, 0}	{#, #, 0, 1, #, 0, 0}
	{#, #, 0, 0, 1, 0, 1}	{#, #, #, 1, 0, 1, 0}

Fig. 4. (A) Performances of the 4096 process-symmetric CAs in the immediate conceptual vicinity of ϕ_{COE2}. The best specimen CA is $\phi_{COE2_{clean}}$ plus one of the combinations of 6 LNC pairs from R.**(B)** E1 schemata prescribing state changes for ϕ_{MM0802}. This is the highest-performing known process-symmetric rule for the DCT.

7 Implicit Evolution of Conceptual Properties?

From the work reported in previous sections, we have established that process symmetry is a conceptual property present in CAs that perform the DCT. Indeed, our experiments have shown that full process symmetry in a high-performing CA ensures that it classifies the two types of IC it encounters equally well. We have also shown that most of the highest-performing CA rules for the DCT are process-symmetric [1].

However, in order to make our results generally useful, i.e. for learning to program cellular arrays that perform a range of tasks that require collective computation, it is important to determine what learning strategy best exploits conceptual properties. For example, CA rules for a different task might not be as amenable to redescription using wildcard schemata (though another type of schema might be appropriate), and they would not necessarily exhibit process symmetry, but perhaps would exhibit other conceptual properties. Therefore, it is important to determine what makes a learning mechanism (e.g. coevolution working with standard CA look-up tables) more likely to exploit conceptual structure during learning most effectively.

In previous work, [2] evaluated learning strategies based on evolution and coevolution, with or without using spatial distribution and local interactions during learning. In particular, they evaluated four methods:

– *Spatial Coevolution*, in which hosts (CA rules) and parasites (ICs) coevolve in a spatial grid in which fitness is calculated and evolutionary selection is done in local grid neighborhoods;

– *Non-spatial Coevolution*, which is the same as spatial coevolution except that fitness calculation and selection are performed using random samples of parasites or hosts that are not spatially correlated;
– *Spatial Evolution*, which uses the same spatial grid method as spatial coevolution, except that the ICs do not evolve but are generated at random at each generation; and
– *Nonspatial Evolution*, which is similar to a traditional genetic algorithm.

Their results have shown that spatial coevolution is substantially more successful than the other methods in producing high-performance rules. [2] gave evidence that this learning strategy ensures the highest diversity in the host (evolving programs) populations, which allows higher-performing CA rules to be discovered.

The preliminary results we report here suggest that this diversity, coupled with the arms-races mechanisms at work during spatial coevolution, leads over time, to the survival of CAs that are more generally capable of solving the two types of IC they encounter. This is illustrated in Figure 5, where the degree of process symmetry (continuous line), and the overall performance (dots) for the best individual in a population during a number of runs for each of the learning strategies is shown[10].

It is clear from the plot in Figure 6 that spatial coevolution has the smallest differences between performances for the two types of IC over time, and that there appears to be a correlation between performance and degree of process symmetry. Moreover, there seems to be sudden changes occurring in some of the plots. In particular for spatial coevolution, these changes show correlated increases in overall performance and degree of process symmetry.

Concerning the apparent correlation between degree of process symmetry and performance, Table 2 shows the Pearson correlation coefficients for the data analyzed and partially plotted in Figure 5. Using 1000 degrees of freedom, with 99.99% confidence (crit. value 0.104), the *Non Spatial Coevolution* strategy has weak negative correlation for the 1st run; no correlation for the 2nd; weak positive correlation for the 3rd; and no correlation for the 4th. The *Non Spatial Evolution* strategy has significant positive correlation for the 1st run; significant negative correlation for the 2nd; and weak negative correlation for the 3rd. The *Spatial Coevolution* strategy has significant positive correlation for the 1st and 3rd runs; weak positive correlation for the 2nd, and very strong positive correlation for the 4th. Lastly, the *Spatial Evolution* strategy has significant positive correlation for the 1st run; for the 2nd and 3rd runs there is a weak positive correlation, and no correlation for the 4th.

Clearly, if process-symmetry is taken to be a learning goal, spatial coevolution appears to be the only strategy capable of achieving this learning. To a lesser degree the spatial evolution strategy can also achieve this, while the non-spatial strategies do not achieve this learning consistently.

[10] Here only two runs for each strategy are plotted for clarity. However, a larger number of runs (mostly four per strategy) was analyzed. Full plots are available from http://mypage.iu.edu/~marquesm/Site/Online_Materials/

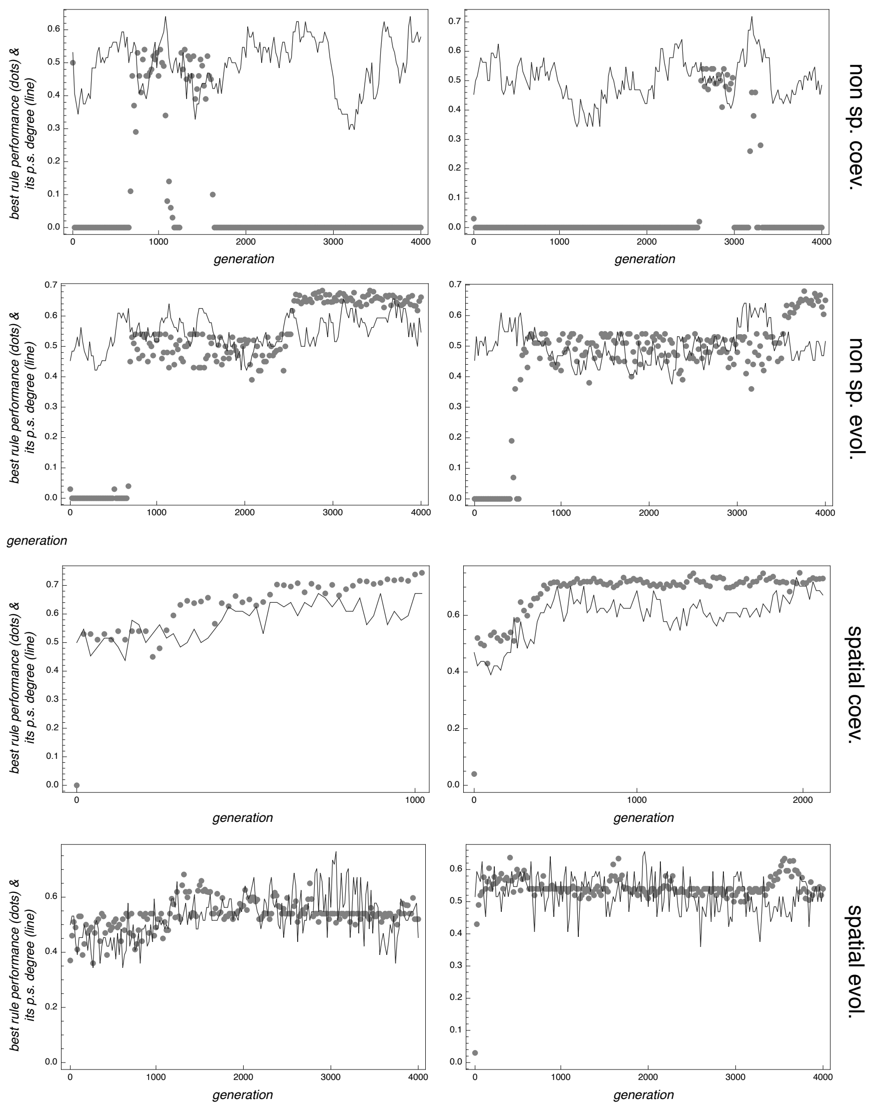


Fig. 5. Average performance of the best individual CA rule in a generation (dots), and its degree of process-symmetry (line) for different runs of four learning strategies to evolve CAs that perform the DCT.

We investigated the apparent sudden changes mentioned earlier (most noticeable in the spatial coevolution plots in Figure 5). Figure 6, shows the same data plotted in Figure 5, but splitting the performance by type of IC. The lighter dots show the rule's performance in classifying correctly cases in which the IC has majority 0s; darker dots show the performance for the converse type of prob-

	Run 1	Run 2	Run 3	Run 4
N.S. Coe	-0.15	0.05	0.11	0.05
N.S. Evo	0.43	-0.48	-0.18	--
SP. Coe	0.62	0.13	0.65	0.8
SP. Evo	0.31	0.17	0.11	0.07

Table 2. Correlation between performance and degree of process-symmetry for each run over evolution strategy.

lem (majority 1s) and the continuous line is the degree of process symmetry. It becomes clear from the figure that, for the spatial coevolution strategy, there is an initial period in which the hosts are very good at solving one type of problem, but very poor at the converse type. After the abrupt change in performance differences per type of IC, in which process-symmetry also increases.

8 Solving the DCT in 2D

In §5 we described a methodology to perform evolutionary search in a space of process symmetric rules, looking for the best process symmetric rule to perform the DCT in one dimension. We applied the same methodology to search the (much larger space containing 2^{2^9} CAs) of rules to perform the DCT in two dimensions, using the Moore neighborhood (center cell and 8 adjacent cells). Instead of looking in the space of 4-wildcards, we searched the space of four, five and six wildcards. In the space of six wildcards our search discovered the highest-performing 2D CA rule for the DCT found to date. The performance of this rule on 2D lattices of 19x19 cells is about 85%. Moreover, Aitana's redescription of this rule $\phi_{MM2D320}$ is very compact (shown in Figure 7), which shows the rule is parsimonious.

9 Conclusions and Future Work

In this paper we have demonstrated that a particular conceptual structure, process symmetry, is correlated with performance on the density classification task. We have also demonstrated that restricting the evolutionary algorithm's search to the space of process-symmetric rules can more easily produce high-performance rules—for both one and two dimensional CAs—than allowing the EA an unrestricted search space.

Furthermore, we have provided evidence that spatial coevolution, previously shown to be a powerful method for evolving cellular automata for the DCT, implicitly increases the degree of process symmetry in CAs over generations, and is correlated with the CAs improvement in performance.

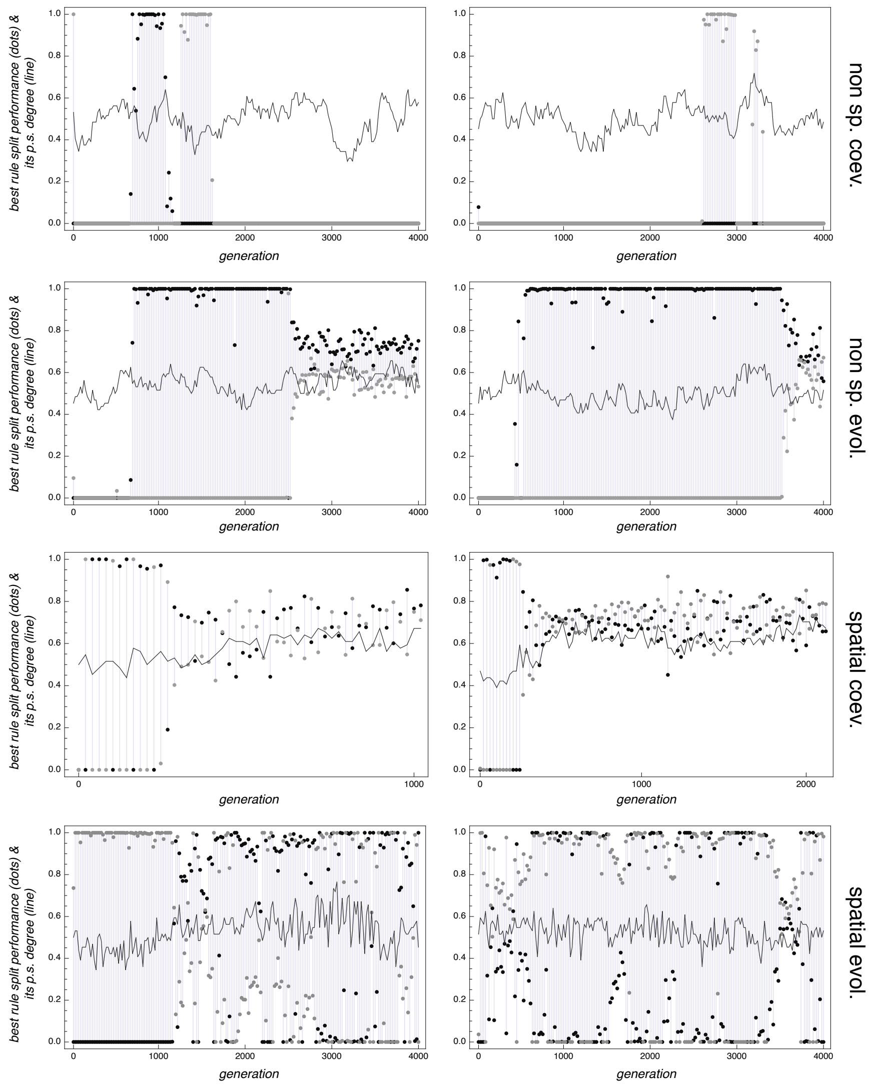


Fig. 6. Split performance of the best individual CA rule in a generation—lighter dots for performance on ICs with majority 0s, darker dots for performance on ICs with majority 1s, and its degree of process-symmetry (line) for different runs of four learning strategies to evolve CAs that perform the DCT. For each best individual CA in a generation, a vertical light-gray line is used to join the two different performances, showing the difference between them.

RULE	Generation	Annihilation

$\phi_{MM2D320}$	{#,#,#,#,0,#,#,1,1}
	{#,#,1,#,0,1,#,#,#}
	{#,1,#,#,0,1,#,#,#}
		{0,0,#,#,1,#,#,#,#}
		{#,#,#,0,1,#,0,#,#}
		{#,#,#,0,1,#,#,0,#}

Fig. 7. E1 of $\phi_{MM2D320}$ The first three elements correspond to the NW, N, NE, W, updating, E, SW, S, and SE neighbors in that order.

The major goals for future work on this topic are (1) determining how well Aitana can discover useful conceptual structures for other, more complex computational tasks for CAs; (2) developing a better understanding of *why* particular conceptual structures such as process symmetry enable higher-performance, and (3) further investigation of the implicit evolution of conceptual structures in CA rule tables, and determining if and how these structures are related to characterizations of the space-time behavior of CAs, such as the domains and particles framework of Crutchfield et al. [18].

In recent work [19] have found new CA rules for the 1-dimensional DCT problem with performances over 88%. Future work is needed in order to determine the split performances of these new, high-performing CAs, as well as their conceptual structure—both in terms of parsimoniousness, and their levels of process symmetry.

10 Acknowledgements

Melanie Mitchell acknowledges support from the Focus Center Research Program (FCRP), Center on Functional Engineered Nano Architectonics (FENA). This work was partially supported by Fundação para a Ciência e a Tecnologia (Portugal) grant 36312/2007.

References

1. Marques-Pita, M., Manurung, R., Pain, H.: Conceptual representations: What do they have to say about the density classification task by cellular automata? In Jost, J., Reed-Tsotchas, F., Schuster, P., eds.: ECCS'06, European Conference on Complex Systems. (2006)
2. Mitchell, M., Thomure, M.D., Williams, N.L.: The role of space in the success of coevolutionary learning. In: Proceedings of Artificial Life X: Tenth Annual Conference on the Simulation and Synthesis of Living Systems. (2006)
3. Zhirnov, V., Cavin, R., Lemming, G., Galatsis, K.: An assessment of integrated digital cellular automata architectures. Computer **41**(1) (2008) 38–44
4. Mitchell, M., Crutchfield, J., Hraber, P.: Revisiting the edge of chaos: Evolving cellular automata to perform computations. Complex Systems **7** (1993) 89–130

5. Gacs, P., Kurdyumov, L., Levin, L.: One-dimensional uniform arrays that wash out finite islands. Probl. Peredachi. Inform. **14** (1978) 92–98
6. Gonzaga de Sá, P., Maes, C.: Gacs-Kurdyumov-Levin automaton revisited. Journal of Statistical Physics **67**(3-4) (1992) 507–522
7. Andre, D., III, F.B., Koza, J.: Discovery by genetic programming of a cellular automata rule that is better than any known rule for the majority classification problem. In Koza, J., Goldberg, D., Fogel, D., eds.: Proceedings of the First Annual Conference on Genetic Programming, MIT Press (1996) 3–11
8. Das, R., Mitchell, M., Crutchfield, J.: A genetic algorithm discovers particle-based computation in cellular automata. In Davidor, Y., Schwefel, H.P., Männer, R., eds.: Proceedings of the Int.Conf. on Evolutionary Computation. (1994) 344–353
9. Juillé, H., Pollack, B.: Coevolving the ideal trainer: Application to discovery of cellular automata rules. In Garzon, M.H., Goldberg, D.E., Iba, H., Riolo, R., eds.: Genetic Programming 1998: Proceedings of the Third Annual Conference, San Francisco, Morgan Kauffmann (1998)
10. Ferreira, C.: Gene expression programming: A new adapive algorithm for solving problems. Complex Systems **13**(2) (2001) 87–129
11. Karmiloff-Smith, A.: Beyond Modularity: A Developmental Perspective on Cognitive Science. MIT Press (1992)
12. Gärdenfors, P.: Conceptual Spaces: The Geometry of Tought. MIT Press/Bradford Books (2000)
13. Marques-Pita, M.: Aitana: A Developmental Cognitive Artifact to Explore the Evolution of Conceptual Representations of Cellular Automata-based Complex Systems. PhD thesis, School of Informatics, University of Edinburgh, Edinburgh, UK (2006)
14. Holland, J., Holyoak, K., Nisbett, R., Thagard, P.: Induction: Processes of Inference, Learning and Discovery. MIT Press (1986)
15. Piaget, J.: The Origins of Intelligence in Children. International University Press (1952)
16. Piaget, J.: The Child's Construction of Reality. Routledge and Kegan Paul (1955)
17. Marques-Pita, M., Rocha, L.M.: Conceptual structure in cellular automata: The density classification task. In Bullock, S., Noble, J., Watson, R.A., Bedau, M.A., eds.: Proceedings of the Eleventh International Conference on Artificial Life (Alife XI), MIT Press, Cambridge, MA. (2008)
18. Crutchfield, J.P., Mitchell, M., Das, R.: The evolutionary design of collective computation in cellular automata. In Crutchfield, J.P., Schuster, P.K., eds.: Evolutionary Dynamics—Exploring the Interplay of Selection, Neutrality, Accident, and Function. Oxford University Press, New York (2003) 361–411
19. Woltz, D., De Oliveira, P.: Very effective evolutionary techniques for searching cellular automata rule spaces. Journal of Cellular Automata (to appear)