A comparison of evolutionary and coevolutionary search

Ludo Pagie* and Melanie Mitchell
Santa Fe Institute
1399 Hyde Park Road
Santa Fe, NM87501, US
ludo@santafe.edu
mm@santafe.edu
phone: (+1)(505)-984-8800
fax: (+1)(505)-982-0565

Abstract

Previous work on coevolutionary search has demonstrated both successful and unsuccessful applications. As a step in explaining what factors lead to success or failure, we present a comparative study of an evolutionary and a coevolutionary search model. In the latter model, strategies for solving a problem coevolve with training cases. We find that the coevolutionary model has a relatively large efficacy: 86 out of 100 (86%) of the simulations produce high quality strategies. In contrast, the evolutionary model has a very low efficacy: a high quality strategy is found in only two out of 100 runs (2%). We show that the increased efficacy in the coevolutionary model results from the direct exploitation of low quality strategies by the population of training cases. We also present evidence that the generality of the high-quality strategies can suffer as a result of this same exploitation.

1 Introduction

Coevolutionary search is an extension of standard evolutionary search in which the fitness of evolving solutions depends on the state of other, coevolving individuals rather than a fixed evaluation function. Coevolutionary search involves either one or two populations. In the first case individual fitness is based on competitions among individuals in the population (Rosin & Belew, 1997; Angeline & Pollack, 1993; Sims, 1994). In the second case the fitness of individuals is based on their behavior in the context of the individuals of the other population (Hillis, 1990; Juillé & Pollack, 1996; Paredis, 1997;

*Current address: Department of Zoology, University of Oxford, South Parks Road, Oxford OX1 3PS, UK

Pagie & Hogeweg, 1997). This latter type of coevolutionary search is often described as "host-parasite", or "predator-prey" coevolution.

The reasons typically cited for the expected increased success and efficiency of coevolutionary search algorithms are the gain in efficiency of the evaluation of evolving solutions (Hillis, 1990), the possible automatic adjustment of the selection gradient which is imposed on the evolving solutions (Juillé & Pollack, 2000), and the potential open-ended nature of coevolutionary systems (Rosin & Belew, 1997; Ficici & Pollack, 1998).

Some successful applications of coevolutionary search have used spatially embedded systems (Hillis, 1990; Husbands, 1994; Pagie & Hogeweg, 1997). In such systems the individuals are distributed on a regular grid and the evolutionary processes (i.e., fitness evaluation, selection, and, if applicable, crossover) take place locally among neighboring individuals. Hillis (1990) and Pagie & Hogeweg (1997) gave examples of spatially embedded coevolutionary models that were more successful than spatially embedded, but otherwise standard, evolutionary models for the same search problem. Other successful applications of coevolution have used techniques such as fitness sharing to produce and maintain diverse populations (Rosin & Belew, 1997) or techniques such as lifetime fitness evaluation to retain good solutions in the populations (Paredis, 1997).

In contrast to the successful applications of coevolutionary search there are a number of studies showing that coevolutionary dynamics can lead to non-successful search as well. Such cases are often characterized by the occurrence of *Red Queen dynamics* (Paredis, 1997; Shapiro, 1998; Pagie & Hogeweg, 2000; Juillé & Pollack, 2000), *speciation* (Pagie, 1999), or *mediocre stable states* (Pollack *et al.*, 1997; Juillé & Pollack, 2000).

In earlier work it was shown how Red Queen dynamics can occur in a spatially embedded coevolutionary model under continuous mixing [*], while high quality solutions evolve in the same model if mixing is omitted so that spatial patterns can form (Pagie & Hogeweg, 2000). In the latter case speciation events can result in persisting sub-species, whereas in the former case global competitive exclusion occurs on a much shorter time-scale and tends to converge the population to one species. Heterogeneity in the populations prevents Red Queen dynamics from persisting. Instead, evolution proceeds to produce general, high quality solutions (Pagie & Hogeweg, 2000), or results in very diverse populations in which the individuals exhibit sub-optimal behavior (Pagie, 1999).

Here we report results from a follow-up comparative study of a coevolutionary and a standard evolutionary search process, both of which are embedded in space. The most significant result of the study is the difference in efficacy of the two models on a given task. Whereas the coevolutionary model finds solutions that use high quality strategies in 86 out of 100 runs (86%), the evolutionary model finds such a solution only twice in 100 runs (2%). By comparing the evolutionary dynamics in the two models we try to characterize the processes that lead to this large difference in search efficacy, thereby pinpointing important aspects of coevolutionary search.

[*] Continuous mixing in a spatial model approximates a non-spatial model in which selection and recombination take place without regard for spatial location of individuals.

2 CA density classification task

The task given to both models is the density classification task for cellular automata (CAs) (Packard, 1988; Mitchell *et al.*, 1994). The density classification task is defined for one-dimensional, binary state CAs with a neighborhood size of 7; that is, at each time step a cell's new state (0 or 1) is determined by its current state and the states of its three neighbors on each side. The CAs employ periodic (circular) boundary conditions.

The task for a CA is to classify bit strings on the basis of density, defined as the fraction of 1s in the string. If the bit string has a majority of 0s it belongs to density class 0; otherwise it belongs to density class 1. Here we restrict our study to bit strings of length 149, which means that the majority is always defined. The CAs in our study operate on a lattice of size 149 whose initial configuration of states is the given bit string. (Note that the performance of a CA on this task generally degrades with increasing length of the bit string.) A CA makes 320 iterations (slightly more than twice the lattice size). If the CA settles into a homogeneous state of all 0s it classifies the bit string as density class 0. If the CA settles into a homogeneous state of all 1s it classifies the bit string as density class 1. If the CA does not settle into a homogeneous state it makes by definition a mis-classification. Land & Belew (1995) showed that no two-state CA exists that can correctly classify all possible bit strings of arbitrary lengths, but did not give an upper bound on possible classification accuracy.

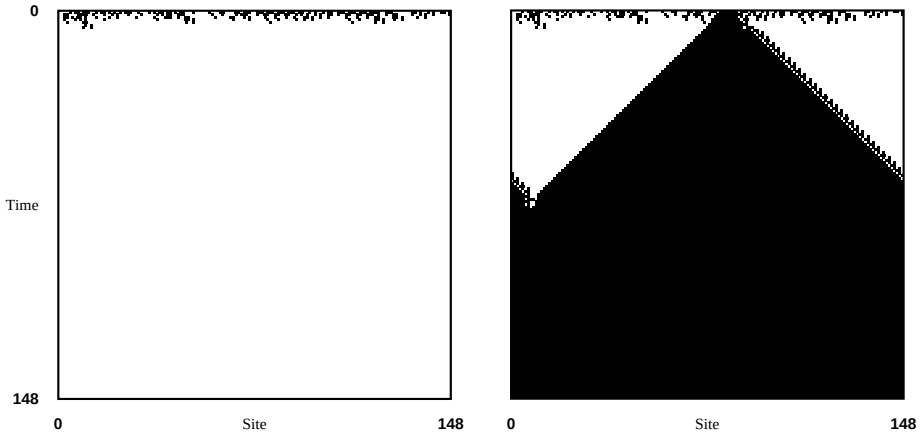

Figure 1: Space-time diagrams illustrating the behavior of a typical block-expanding strategy. The 149-cell one-dimensional CA lattice is displayed on the horizontal axis, with time increasing down the page. Cells in state 0 are colored white; cells in state 1 are colored black. The initial configuration in the diagram on the left has low density and is correctly classified as class 0. The initial configuration in the diagram on the right has high density. It contains a sufficiently large block of 1s, which expands to fill up the lattice, resulting in a correct classification as class 1. (Adapted from Mitchell *et al.*, 1996.)

An objective measure of the classification accuracy of a CA is its *performance* value

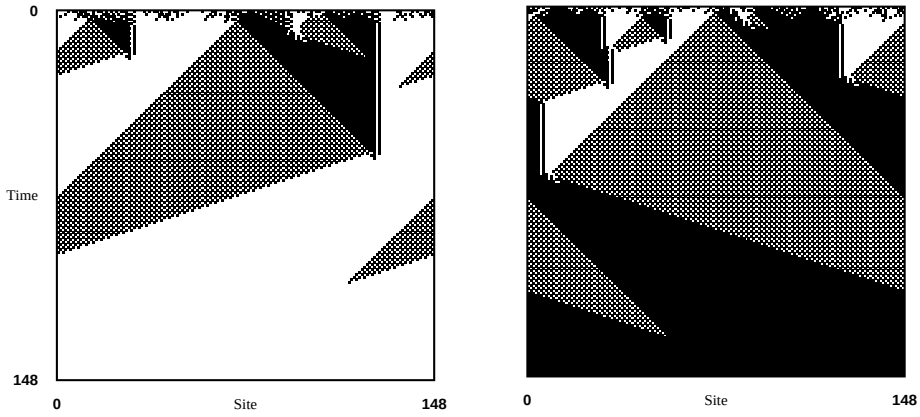

Figure 2: (a) Space-time diagram illustrating the behavior of a typical particle strategy. The initial configuration in the diagram on the left has low density and the correct classification of all 0s. The high-density initial configuration on the right is also correctly classified. (Adapted from Mitchell *et al.*, 1996.)

P, defined as the fraction of correct classifications of 10^4 strings, each selected from an "unbiased" distribution—a binomial density distribution centered around density 0.5. (This distribution results in strings with density very close to 0.5—the most difficulty cases to correctly classify.) An important characteristic of a CA is the type of strategy that it uses in order to classify a string. The three main strategies discussed by Mitchell *et al.* (1994)—all discovered by their genetic algorithm— are default, block-expanding, and particle. The default strategy classifies all bit strings to a single density class (class 0 or class 1) by always iterating to a homogeneous state of all 0s or all 1s. Thus there are two types of default strategies; default-0 and default-1. Default strategies have performance values of approximately 0.5, since approximately half of a random sample of 10^4 strings will be classified correctly. Block-expanding strategies, illustrated in fig. 1, are refinements of the default strategy. In a block-expanding strategy bit strings are classified to one density class, for instance density class 0 (left-hand side of fig. 1), unless the bit string contains a sufficiently large block of, in this case, 1s, which results in the block expanding to cover the whole lattice, yielding a classification of density class 1 (right-hand side of fig. 1). The definition of "sufficiently large" block depends on the particular CA, but typically means blocks equal to or larger than the CA neighborhood size of 7 cells. This "strategy" reflects the fact that the presence of a block of 1s (0s) in a 149-bit string roughly correlates with overall high (low) density, and the string is classified accordingly. Block-expanding strategies typically have performance values between 0.55 and 0.72 on bit strings of length 149. Finally, particle strategies, illustrated in fig. 2, use interactions between meso-scale patterns in order to classify bit strings (Crutchfield & Mitchell, 1995; Das *et al.*, 1994). They typically have performance values of 0.72 and higher on bit strings of length 149.

The three classification strategies, default, block-expanding, and particle, exhibit increasing computational complexity (Crutchfield & Mitchell, 1995). In evolutionary

search runs one typically sees a series of epochs in which default strategies are evolved first, then block-expanding strategies, and finally (on some runs) particle strategies (Crutchfield & Mitchell, 1995). The highest observed performance to date of a particle CA on the density-classification task is 0.86 on bit strings of length 149 (Juillé & Pollack, 2000).

A number of research groups have used evolutionary search to find strategies for the CA density classification task as a paradigm for the evolution of collective computation in locally interacting dynamical systems (e.g., Mitchell *et al.*, 1996; Juillé & Pollack, 2000). In a standard evolutionary setup, the fitness of a CA is the fraction of correct classifications it makes on a small random sample of bit strings interpreted as initial configurations (ICs) for the CA. In a coevolutionary setup the bit strings, or ICs, make up the second population and the fitness evaluation of the CAs is based on the evolving ICs. Coevolutionary models which focus on this task were previously studied by Paredis (1997), Juillé & Pollack (2000), and Pagie & Hogeweg (2000).

Previous studies indicate that, under a standard evolutionary model as well as previous coevolutionary models, it is difficult to evolve high-performance CAs; only a small number of runs produce CAs that use particle strategies. If we define the *efficacy* of an evolutionary search model as the percent of runs of that model that produce particle strategies, then the efficacy of standard evolutionary and previous coevolutionary search models has been found to fall somewhere between 0% and 40% (Mitchell *et al.*, 1996; Werfel *et al.*, 2000; Juillé & Pollack, 2000; Oliveira *et al.*, 2000), with the remaining runs producing block-expanding CAs (see Mitchell *et al.*, 1994 for some comments on this issue). Two studies report very high efficacy (Andre *et al.*, 1996; Juillé & Pollack, 2000) but in these cases the computational resources used per simulation were orders of magnitude higher than used here or in the other studies.

Coevolutionary models applied to the CA density classification task generally show Red Queen dynamics of the CAs and ICs (Paredis, 1997; Pagie & Hogeweg, 2000; Juillé & Pollack, 2000). For the density classification task this type of evolutionary dynamics is characterized by oscillations in the types of ICs and CAs that are present in the population. The IC population oscillates, with a fairly constant frequency, between bit strings with density less than 0.5 and bit strings with density greater than 0.5. The CA population oscillates at a similar frequency between default-0 strategies, which correctly classify the former, and default-1 strategies, which correctly classify the latter. Paredis (1997) circumvented Red Queen dynamics by replacing the coevolution of the ICs by randomly generated ICs. Juillé & Pollack (2000) changed the coevolutionary setup by introducing a limit on the selection of ICs such that it was constrained by the ability of the CAs. Pagie & Hogeweg (2000) embedded the coevolutionary model in a 2D grid and introduced an extension on the fitness function used to evaluate the ICs which imposes stabilizing selection on the ICs toward strings that can be classified more easily by CAs. Here we use the same IC fitness function $\Phi(IC_i)$:

$$\Phi(IC_i) = \begin{cases} 0 & \text{if classified correctly} \\ |density(IC_i) - \frac{1}{2}| & \text{otherwise.} \end{cases}$$

3 The model

The populations in both the evolutionary and coevolutionary models are embedded in a two-dimensional regular grid of 30 by 30 sites with periodic boundary conditions. At each location in this grid one CA and one IC are present, giving 900 individuals in each population. In the coevolutionary model both CAs and ICs evolve whereas in the evolutionary model only the CAs evolve while the ICs are generated anew every generation according to a fixed procedure (see below).

parameter	value
grid size	30×30
population sizes	900
generations per run	5000
CA mutation probability	0.0016
IC mutation probability	0.0034

Table 1: The model parameters.

The fitness of a CA is based on the nine ICs in its Moore neighborhood (i.e., the same site and the eight adjoining neighbors). The fitness of an IC is based only on the CA in the same site. This asymmetric fitness evaluation was found to improve the evolutionary search process (Pagie & Hogeweg, 1997). In both models the fitness evaluation is extremely sparse. Sparse evaluation is in fact unavoidable since a complete evaluation of a CA would cover all 2^{149} possible ICs—clearly an infeasible task. Pagie & Hogeweg (1997) showed that sparse fitness evaluation can in fact help the evolutionary process rather than hinder it (see also Hillis, 1990 for a discussion of sparse evaluation).

Selection of CAs in both models is based on tournament selection. Each CA in the population is replaced by the winner of the tournament including the CA itself and its eight neighboring CAs. The winner is selected, probabilistically, based on the rank order of the fitness of individuals in the tournament. The probability of an individual to be selected is 0.5^{rank}, for the eight highest ranked individuals. The lowest ranked individual (i.e., *rank*=9) has a probability 0.5^8 of being selected, so that the sum of all the probabilities will be 1. In the coevolutionary model the same selection procedure is applied to the ICs.

After selection we apply point mutations to the CAs and, in the coevolutionary model, to the ICs. The mutation probabilities are 0.0016 per bit for the CAs and, in the coevolutionary case, 0.0034 per bit for the ICs[†]. For the sake of simplicity in this study, crossover is not used in either model. The parameters that we used in the model are summarized in table 3.

In the evolutionary model the population of ICs is generated anew at every generation, selected at random from a uniform density distribution in which each density in [0,1] is equally likely (c.f. Werfel *et al.*, 2000, see Mitchell *et al.*, 1994; Juillé &

[†]The mutation probabilities that we used used correspond to an expected number of mutation events of $0.2 \times$ population size in the CA population and $0.5 \times$ population size in the IC population.

Pollack, 2000 for a discussion on the effect of using a uniform versus a binomial distribution over densities). Thus, the only difference between the two models is the origin of the new generations of ICs: based on the previous generation in the coevolutionary model and based on a fixed procedure in the evolutionary model.

The CAs are initialized as random bit strings with an unbiased (binomial) density distribution. In the coevolutionary model the ICs are initialized with all-0 bit strings. This initialization of the ICs strongly induces a transient phase of Red Queen dynamics which, in a spatially embedded model, is unstable (Pagie & Hogeweg, 2000). Each simulation is run for 5000 generations. Although this number is high compared to the length of evolutionary runs in most other studies, the computational burden per generation in our model is relatively small because each CA is evaluated on only nine ICs. Define an *IC evaluation* as the classification by a CA of a single IC. Then in each model, at each generation $900 \times 9 = 8100$ IC evaluations are performed, i.e., 4.05×10^7 IC evaluations per simulation. This compares to a total number of IC evaluations per simulation in other studies as: 2.61×10^9 in Andre *et al.* (1996), 10^6 in (Das *et al.*, 1994) and Oliveira *et al.* (2000), 4.0×10^7 in Werfel *et al.* (2000), and 5.0×10^9 and 4.0×10^7 (in two different sets of runs) in Juillé & Pollack (2000). For each model we ran 100 simulations each starting with a different random seed.

4 Results

4.1 Efficacy

Table 2 lists the efficacy for each model and each classification strategy, i.e., the number of runs out of 100 on which the given strategy was produced. Each model produced both default and block-expanding strategies on every run. Eighty-six coevolutionary runs produced particle CAs, whereas only two evolutionary runs did.

model	default	block	particle
coevolution	100	100	86
evolution	100	100	2
modified coevolution	100	100	23

Table 2: Number of runs (out of 100 total runs) in which each of the three strategies is found in the coevolutionary model, the evolutionary model, and in a modified version of the coevolutionary model (see sect. 4.4).

Figure 3 plots, for each of the 100 runs, the maximum performance P obtained by a CA in that run, for both the coevolutionary model (solid line) and evolutionary model (dashed line). The values are ordered according to decreasing performance values. The symbols denote which strategy is used by the CAs: block-expanding ($\bigtriangledown$) or particle ($\bigtriangleup$).

The difference in efficacy between the coevolutionary and the evolutionary models is very striking. The evolutionary model is unable in almost all cases to make the transition from the block-expanding strategy to the particle strategy, whereas the coevolutionary model is able to make this transition in most runs. We have identified a

mechanism that takes place in the coevolutionary model—and not in the evolutionary model—that we believe underlies its increased efficacy: the evolution of bit patterns in the ICs that specifically target block-expanding CAs. (There are likely to be additional mechanisms that we have not yet identified.) We describe this mechanism and some of its side effects in the next subsections.

4.2 Coevolution of "deceptive" blocks

As was discussed in section 2, the block-expanding strategy operates by expanding sufficiently large blocks of 1s (or 0s) in the IC to produce an all-1 (all-0) state. This strategy can produce fairly high *fitness* when the fitness of a CA is evaluated using random ICs drawn from a uniform distribution, as is done in the evolutionary model, but produces much lower *performance* P, which is evaluated using random ICs drawn from the unbiased (binomial) distribution.

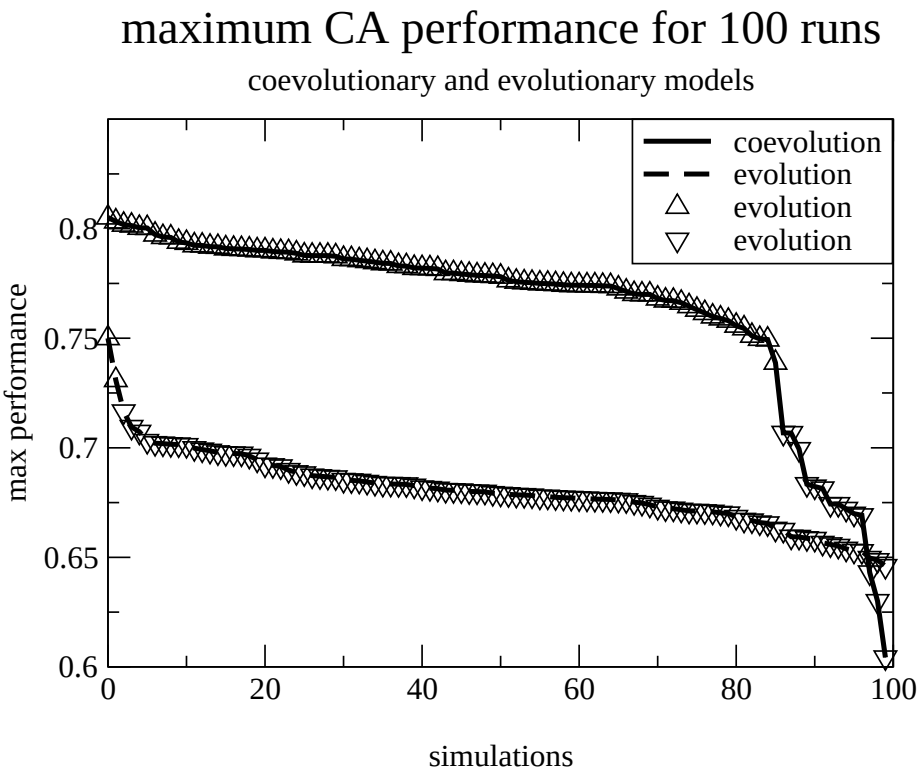

Figure 3: The maximum CA performance P found for each of the 100 runs in the coevolutionary model (solid line) and the evolutionary model (dashed line). The coevolutionary model produces high-performance particle strategies ($\triangle$) more often than the evolutionary model which produces lower-performance block-expanding strategies ($\triangledown$) more often.

In the coevolutionary model, block-expanding strategies can easily be exploited by coevolving ICs, and this process of exploitation is very important for producing the high efficacy seen in the coevolutionary model. An IC with a low (high) density can incorporate a block of 1s (0s) which deceptively indicates a high (low) density to a block-expanding CA and thereby force mis-classification. We call such blocks *deceptive blocks*.

For the purpose of this investigation we define the occurrence of a "deceptive block" in an IC to be the occurrence of a block of seven or more adjacent 0s or 1s in the 149-bit-string but only when the probability of the occurrence of such a block in a randomly generated string of the same density is less than 0.01. The top panel of fig. 4 illustrates the occurrence and dynamics of deceptive blocks in the IC population in a run of the coevolutionary model. We plot, at every 10 generations, the proportion of ICs in the population that contain deceptive blocks as defined above. We have marked the periods during the simulation when the highest performance CAs use default strategies, block-expanding strategies, and particle strategies. Clearly, in the period when the CAs use block-expanding strategies the number of ICs that contain deceptive blocks is significantly higher than would be expected in random bit strings of the same density. These deceptive blocks—a result of coevolution that directly targets the block-expanding CAs in the population—push the CA population to discover new, more sophisticated strategies that are immune to deceptive blocks.

4.3 Coevolution of bit patterns that target weaknesses in particle CAs

The effect of the coevolution of particular bit patterns in the (coevolving) ICs becomes clear when we compare the classification accuracy of CAs based on evolved ICs and based on random ICs with identical density values. Evolved ICs in many cases contain bit patterns that exploit certain weaknesses in CA strategies and thus lead to reduced classification accuracy of the CAs. The random ICs are expected not to contain such bit patterns; the classification accuracy of CAs based on these ICs serves as an accuracy base-line. The difference in classification accuracy D of a CA is defined as the fraction P_{ran} of correct classifications the CA makes on a set of random ICs minus the fraction P_{evol} of correct classifications it makes on the evolved ICs in a given generation:

$$D = P_{\text{ran}} - P_{\text{evol}}.$$

A non-zero D indicates that the evolved ICs contain bit-patterns that the CA uses as a classification criterion. A negative D indicates that the CA correctly correlates these bit-patterns with density values of the set of evolved ICs. A positive D indicates that the bit-patterns in the set of evolved ICs are incorrectly correlated with the density values.

We have calculated D for concurrent populations of CAs and ICs in a single run. The black curve in the lower panel in fig. 4 plots for every 10th generation the average, $\overline{D}$, of D over all unique CAs in the population. At each generation, for each CA in the population, P_{evol} is calculated using the set of all unique ICs in the concurrent IC population, and P_{ran} is calculated using a randomly generated set of ICs of the same

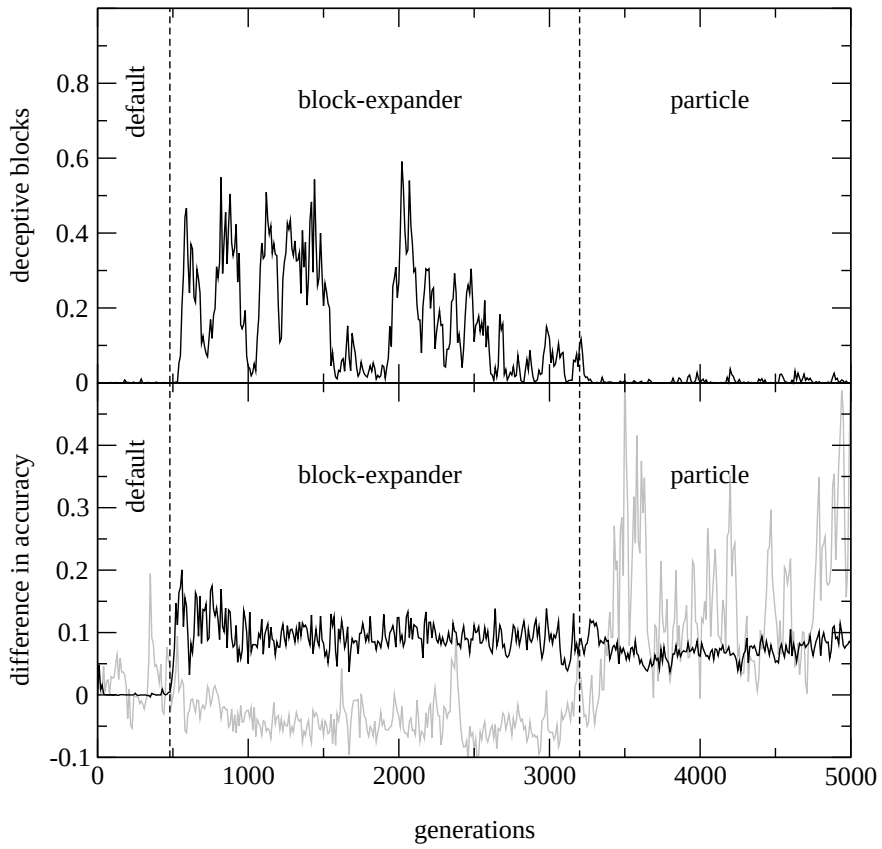

Figure 4: Coevolving ICs exploit CAs. In the top panel the proportion of ICs with deceptive blocks is plotted for a single coevolutionary simulation. When CAs use block-expanding strategies ICs tend to contain significant numbers of deceptive blocks. In the lower panel the black curve plots the average difference in classification accuracy $\overline{D}$ when CAs are evaluated on the basis of evolved ICs and of ICs with equal density values but random bit-patterns. A negative $\overline{D}$ indicates that the CAs base their classification on the presence of particular bit-patterns in the ICs, which in this case are correctly correlated to the density of ICs. A positive $\overline{D}$ also indicates that the CAs base their classification on bit-patterns contained in the ICs but now the bit-patterns are actually correlated to the other density class; the ICs exploit the criterion that the CAs use as a basis for their classification. The grey curve plots $\overline{D}_{\text{fixed}}$, the difference in classification accuracy for a fixed set of five particle CAs from generation 5000 using the same sequence of ICs as was used to calculate $\overline{D}$.

size and with the same densities as the set of evolved ICs. As can be seen in the figure, after the CAs evolve block-expanding strategies, $\overline{D}$ becomes positive and remains so for the rest of the simulation. Thus, in every generation the evolved ICs appear to contain bit patterns such as blocks of 0s or 1s (not present in the random ICs) that are used by the CAs as a basis for their classification. The fact that $\overline{D}$ is positive indicates that CAs are constantly exploited (i.e., deceived) by the bit patterns in the evolved ICs.

We also measure $\overline{D}_{\text{fixed}}$ over a fixed set of CAs, using the same sequence of sets of ICs to calculate P_{evol} and P_{ran} at each generation as were used for $\overline{D}$. The fixed set of CAs consists of five CAs from generation 5000 of this run that use particle strategies and have a relatively high performance value. $\overline{D}_{\text{fixed}}$ is plotted with the grey curve in fig. 4. Surprisingly, $\overline{D}_{\text{fixed}}$ is negative during the block-expander period, which indicates that these five particle CAs exploit the presence of deceptive blocks in ICs; they perform better on the basis of ICs that evolved during the block-expander period than on the corresponding random ICs. However, $\overline{D}_{\text{fixed}}$ becomes sharply positive during the particle period, indicating that these five particle CAs are exploited by the ICs that evolve during this period. In fact, $\overline{D}_{\text{fixed}}$ is even larger during this period than $\overline{D}$, indicating that the particle CAs from generation 5000 are exploited by the evolving ICs more strongly than the particle CAs in earlier generations.

In summary, the plots in fig. 4 indicate that the ICs indeed evolve such that they specifically exploit the weaknesses in the strategies used by CAs, e.g., by incorporating deceptive blocks. This aspect of the coevolution of the CAs and the ICs is very important for the eventual discovery and evolution of particle CAs, and thus the high efficacy of the coevolutionary model. The data suggest that the particle CAs are also exploited by the ICs in a similar manner to the block-expanding CAs. In the next section we will introduce a modification in the coevolutionary model that points out some side effects of this exploitation of particle CAs by the ICs.

4.4 Results from a modified coevolutionary model

To further explore the effect of the evolution of particular bit patterns in the ICs, we modified the coevolutionary model so that no such patterns can form. In this modified model, the coevolving population of ICs contains density values rather than explicit bit strings. At each generation, for each density value in the IC population, we create a new bit string with that density (c.f. Werfel *et al.*, 2000, and Juillé & Pollack, 2000). The mutation of the density values is now defined as a unit change (i.e., $\pm \frac{1}{149}$) in the density value. The mutation rate is such that the same number of mutation events occur in the population of ICs as in the original coevolutionary model (i.e., $0.5\times$ IC population size).

The third row in table 2 shows the efficacy of this modified model. In 23 out of 100 simulations (23%) at least one particle CA was found. The efficacy is much lower than in the original coevolutionary model, which is in accordance with the idea that particle strategies evolve as a consequence of the exploitation of block-expander strategies by the ICs. In the modified model such bit-patterns can no longer be selected for. In comparison with the evolutionary model, however, this modified coevolutionary model still shows a higher efficacy. Apparently, the exploitation of CA strategies by evolving particular bit patterns is not the only mechanism that results in the high efficacy of the

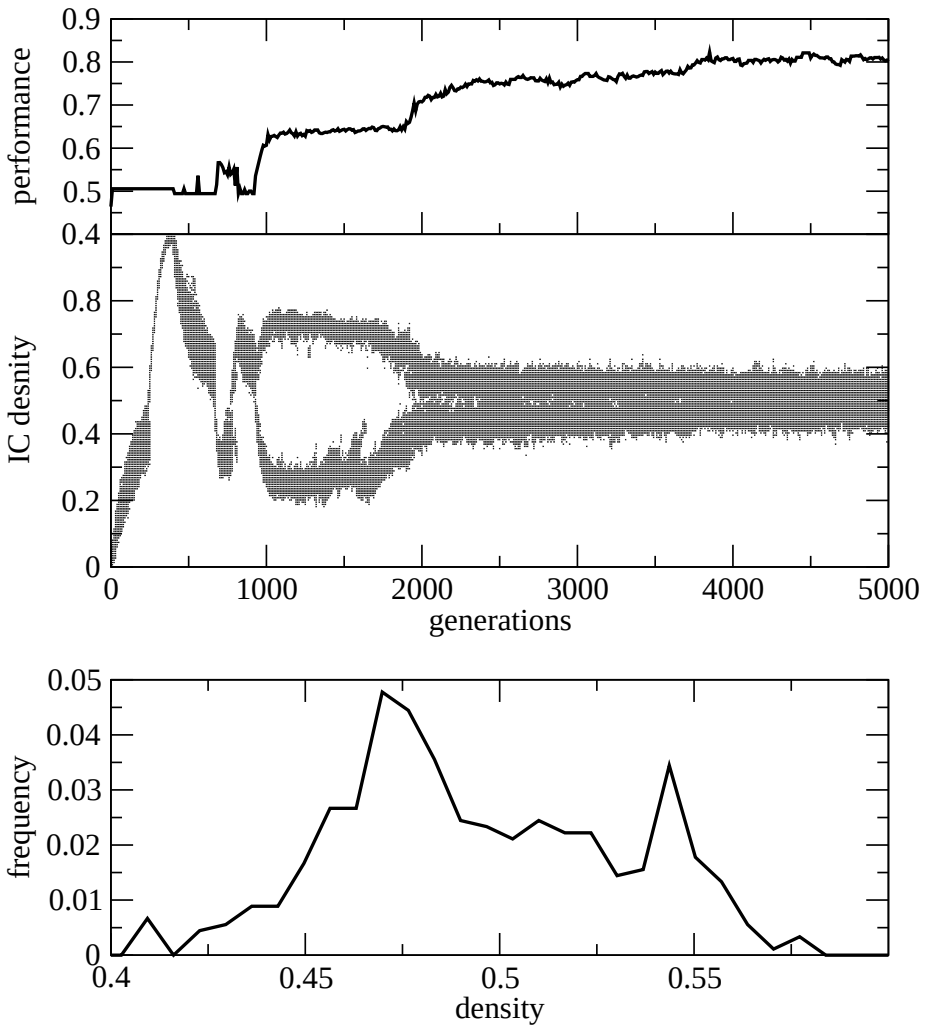

Figure 5: The middle panel shows the evolution of the density distribution of the ICs in a typical run of the modified coevolutionary model. The top panel shows the evolution of the maximum performance value. The lower panel shows the IC density distribution at generation 5000.

coevolutionary model.

An additional mechanism that may be important for the increased efficacy of the modified model compared to the evolutionary model is the change of the distribution of density values of the ICs over time. The middle panel in fig. 5 shows the evolution

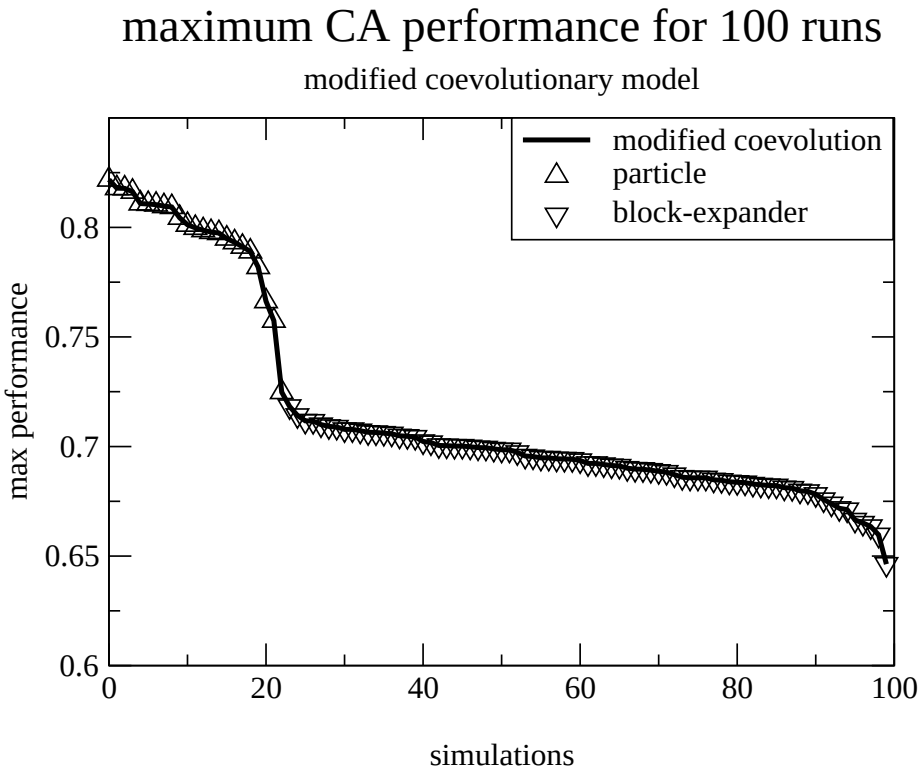

Figure 6: The maximum CA performance found for each of the 100 runs in the modified coevolutionary model. Although the efficacy of the modified model drops sharply, the performance of particle CAs that evolve in this model tend to be somewhat higher than that of the particle CAs that evolve in the original coevolutionary model.

of the IC density distribution in a typical simulation of the modified model; every density in the IC population is plotted at each generation. The upper panel shows the evolution of the maximum CA performance value as context. The lower panel shows the density distribution for generation 5000. The IC population adapts, as can be seen in the changing density distribution, as CAs evolve more sophisticated classification strategies. As a result, initially ICs are not too difficult for evolution to progress; later they become difficult enough in order to present continued selection pressure for the CAs. The density distribution at t=5000 shows two peaks at approximately 0.48 and 0.55. This bimodal distribution indicates the existence of separate sub-species in the IC population (Pagie & Hogeweg, 2000). The coexistence of separate sub-species can be seen even more clearly in the middle panel between $t \approx 1000$ and $t \approx 2000$. Previous work showed that the distribution of IC density values is very important in the evolution of high performance strategies (Mitchell *et al.*, 1994; Juillé & Pollack, 2000).

The particle CAs that evolve in the modified model tend to have higher performance values than the particle CAs that evolve in the original coevolutionary model. In fig.

6 we show the maximum performance values found per simulation for the modified coevolutionary model. In the modified model the average of the highest performance particle CAs that are found during a simulations is 0.80 (stdev=0.02). In the original coevolutionary model the average is 0.76 (stdev=0.04).

The difference of performance values reflects the strong selection bias imposed by the ICs on the CAs in the original model, which is absent in the modified model. In the original model the ICs evolve particular bit patterns which exploit weaknesses in the CAs (fig. 4) and thus present very atypical bit strings to the CAs. In the modified model ICs represent density values only and actual bit strings are made anew every generation. Not only are CAs evaluated on a larger variety of bit strings (ICs are re-initialized every generation and ICs with identical genotypes usually give rise to different bit strings), the bit strings are also initialized randomly, resulting in the presentation of much more generic sample of bit strings during the CA fitness evaluation. Thus it seems that the CAs in the modified coevolutionary model evolved to be more general than those in the original coevolutionary model; the latter evolved to target more specific challenges reflected in the evolving IC population. This may explain the difference in average performance values between the two models.

4.5 Maintenance of diversity

As a consequence of the exploitation of the CAs by the ICs the diversity of strategies that is present in a population in the coevolutionary model is much higher than in the evolutionary model. In the coevolutionary model, at all generations, all possible strategies of equal and lesser quality than the current best strategy are present in the population (fig. 7, top panel). Not only do default CAs coexist with block-expanding CAs, during the block-expanding phase often both types of block-expanding strategies and both types of default strategies are present in the population, or the different types of each strategy are present in an oscillatory manner. In the evolutionary model only during the transient following the (random) initialization do the default-0 and default-1 strategies coexist in the population. As soon as a CA with a higher-than-average performance evolves, here at $t \approx 100$, the whole population is overtaken and the loss of diversity is never reversed (fig. 7, middle panel). In the modified coevolutionary model the diversity is in-between the diversity in the coevolutionary and the evolutionary models (fig. 7, lower panel). The modified coevolutionary model also shows that the CAs with a relatively high performance value tend to dominate the population whereas this is much less the case in the original coevolutionary model. Also, when strategies of a higher quality evolve they tend to dominate the population more strongly than is the case in the coevolutionary model (i.e., at $t \approx 400$ block-expanding begin to dominate and at $t \approx 2800$ particle strategies begin to dominate). A high diversity is often cited as being beneficiary for evolutionary search models. It is unclear whether the difference in diversity that we find here is a side-effect of the evolutionary dynamics or whether it has a causal effect on the efficacy or on the difference in performance values of the evolved particle CAs.

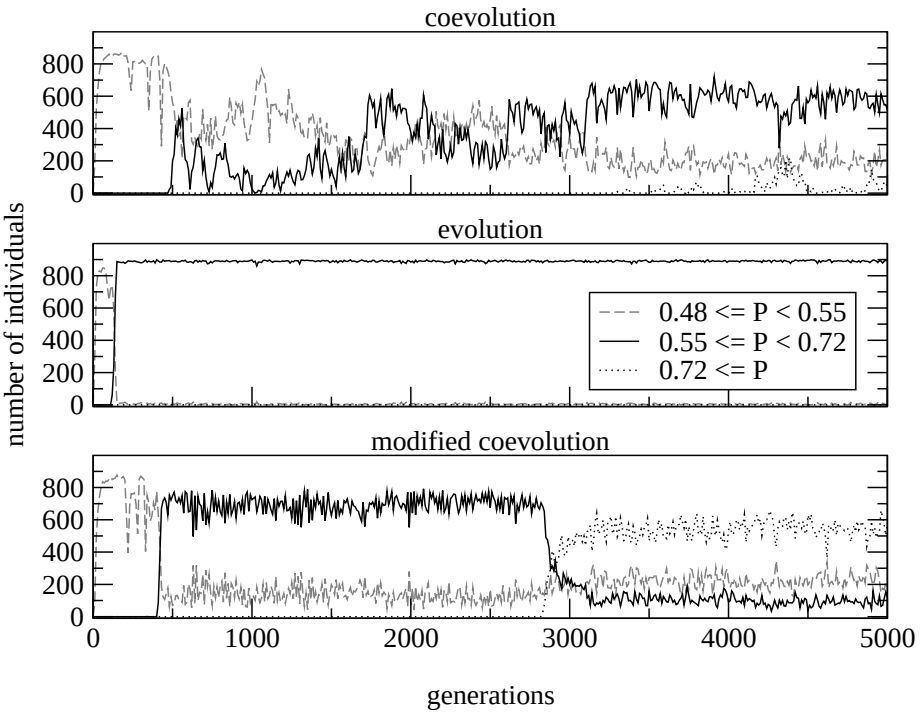

Figure 7: The number of individuals that have a performance value in one of three ranges, in single runs of each of the three models. The performance range gives an indication of strategies that are used by the CAs: $0.48 \leqq P < 0.55$ indicates default strategies; $0.55 \leqq P < 0.72$ indicates block-expanding and low performance particle strategies; $0.72 \leqq P$ indicates particle strategies.

5 Discussion

Coevolutionary search is sometimes believed to lead to arms-races between the co-evolving populations which should lead to the evolution of individuals that exhibit general solutions to a given problem (Rosin & Belew, 1997; Pollack *et al.*, 1997; Paredis, 1997). However, in addition to the evolution of general solutions, coevolution can also result in Red Queen dynamics (Paredis, 1997; Shapiro, 1998; Pagie & Hogeweg, 2000; Juillé & Pollack, 2000), speciation (Pagie, 1999), or mediocre stable states (Pollack *et al.*, 1997; Juillé & Pollack, 2000). It is often unclear under what conditions coevolution will lead to which of these outcomes.

In this paper we compared evolutionary and coevolutionary search for solutions to the the density classification task for cellular automata. We showed that the coevolutionary search model exhibits a very high efficacy: 86% of the runs produce CAs

that use high quality strategies. The standard evolutionary model does so in only one run. We showed that the difference in the efficacy between the two models results from the exploitation by the coevolving population (the ICs) of lower quality strategies, i.e. block-expander strategies. When the ability of the ICs to exploit the CAs is removed, the efficacy of the coevolutionary model drops sharply to 23%.

We have shown that the coevolving IC population explicitly targets weaknesses in the strategies that are used by the CAs. As a result, the CAs undergo strong selection to adapt to this exploitation, which leads to the high efficacy in the coevolutionary model. Our results also suggest that this exploitation can lead to the evolution of less generally applicable CAs, i.e., CAs with lower performance values, than when the exploitation by ICs is prevented (as in the modified coevolutionary model). Interestingly, the original and modified coevolutionary models each has positive and negative effects on the search of optimal CAs by imposing different selection pressures on the CAs. One choice improves the efficacy of the search procedure, while the other choice improves the maximum performance value being evolved.

In summary, the results presented here suggest that coevolution can significantly improve the results of evolutionary search. They also show that there can be a trade-off between overall search efficacy and maximum performance of the evolved individuals. Thus the best method of coevolution to be used on a particular search problem depends on the search goals of the user.

Acknowledgments

This work was supported by the Santa Fe Institute under its core funding and National Science Foundation grant IRI-9705830, and by the Biophysics Group at Los Alamos National Laboratory.

References

ANDRE, D.; BENNETT III, F. H. & KOZA, J. R. (1996); Evolution of intricate long-distance communication signals in cellular automata using genetic programming. In: *Proceedings Artificial Life V*. MIT Press, Nara, Japan.

ANGELINE, P. J. & POLLACK, J. B. (1993); Competitive environments evolve better solutions for complex tasks. In: Forrest, S. (ed.), *Proceedings ICGA 5*. Morgan Kaufmann, pp. 264–270.

CRUTCHFIELD, J. P. & MITCHELL, M. (1995); The evolution of emergent computation. *Proceedings of the National Academy of Sciences* **92**:10742–10746.

DAS, R.; MITCHELL, M. & CRUTCHFIELD, J. P. (1994); A genetic algorithm discovers particle-based computation in cellular automata. In: Davidor, Y.; Schwefel, H.-P. & Männer, R. (eds.), *PPSN III*. Springer-Verlag, vol. 866 of *LNCS*, pp. 344–353.

FICICI, S. G. & POLLACK, J. B. (1998); Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In: Adami, C.; Belew, R. K.; Kitano, H. & Taylor, C. (eds.), *Proceedings Artificial Life 6*. MIT Press, pp. 238–247.

HILLIS, D. W. (1990); Co-evolving parasites improve simulated evolution in an optimization procedure. *Physica D* **42**:228–234.

HUSBANDS, P. (1994); Distributed coevolutionary genetic algorithms for multi-criteria and multi-constraint optimisation. In: Fogarty, T. (ed.), *Evolutionary Computing, AISB Workshop Selected Papers*. Springer-Verlag, vol. 865 of *LNCS*, pp. 150–165.

JUILLÉ, H. & POLLACK, J. B. (1996); Co-evolving intertwined spirals. In: Fogel, L. J.; Angeline, P. J. & Baeck, T. (eds.), *Evolutionary Programming V: Proceedings of the Fifth Annual Conference on Evolutionary Programming*. MIT Press, pp. 461–467.

JUILLÉ, H. & POLLACK, J. B. (2000); Coevolutionary learning and the design of complex systems. *Advances in Complex Systems* **2**(4):371–394.

LAND, M. & BELEW, R. K. (1995); No perfect two-state cellular automata for density classification exists. *Physical Review Letters* **74**(25):5148–5150.

MITCHELL, M.; CRUTCHFIELD, J. P. & DAS, R. (1996); Evolving cellular automata with genetic algorithms: A review of recent work. In: *Proceedings of the First Int. Conf. on Evolutionary Computation and its Applications (EvCA'96)*.

MITCHELL, M.; CRUTCHFIELD, J. P. & HRABER, P. T. (1994); Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D* **75**:361–391. SFI Working Paper 93-11-071.

OLIVEIRA, G. M. B.; DE OLIVEIRA, P. P. B. & OMAR, N. (2000); Evolving solutions of the density classification task in 1d cellular automata, guided by parameters that estimate their dynamic behavior. In: M.A. Bedau, J.S. McCaskill, N. P. & Rasmussen, S. (eds.), *Artificial Life VII*. MIT Press, pp. 428–436.

PACKARD, N. (1988); Adaptation towards the edge of chaos. In: Kelso, J. A. S.; Mandell, A. J. & Shlesinger, M. F. (eds.), *Dynamic Patterns in Complex Systems*, World Scientific. pp. 293–301.

PAGIE, L. (1999); Coevolutionary dynamics: information integration, speciation, and red queen dynamics. In: *Information integration in evolutionary processes*, Bioinformatics group, Utrecht University, www.santafe.edu/˜ludo/articles/thesis.pdf, chap. 5. pp. 67–93.

PAGIE, L. & HOGEWEG, P. (1997); Evolutionary consequences of coevolving targets. *Evolutionary Computation* **5**(4):401–418.

PAGIE, L. & HOGEWEG, P. (2000); Information integration and red queen dynamics in coevolutionary optimization. In: *Proceedings CEC 2000*. pp. 1260–1267.

PAREDIS, J. (1997); Coevolving cellular automata: be aware of the red queen! In: Baeck, T. (ed.), *Proceedings ICGA VII*. pp. 393–400.

POLLACK, J. B.; BLAIR, A. D. & LAND, M. (1997); Coevolution of a backgammon player. In: Langton, C. G. & Shimohara, K. (eds.), *Artificial Life V*. The MIT Press, pp. 92–98.

ROSIN, C. D. & BELEW, R. K. (1997); New methods for competitive coevolution. *Evolutionary Computation* **5**(1):1–29.

SHAPIRO, J. L. (1998); Does data-model co-evolution improve generalization performance of evolving learners? In: Eiben, A.; Bck, T.; Schoenauer, M. & Schwefel, H.-P. (eds.), *PPSN V*. vol. 1498 of *LNCS*, pp. 540 – 549.

SIMS, K. (1994); Evolving 3D morphology and behavior by competition. In: Brooks, R. A. & Maes, P. (eds.), *Proceedings Artificial Life IV*. MIT Press, pp. 28–39.

WERFEL, J.; MITCHELL, M. & CRUTCHFIELD, J. P. (2000); Resource sharing and coevolution in evolving cellular automata. *IEEE Transactions on Evolutionary Computation* **4**(4):388–393.