# Pareto Optimality in Coevolutionary Learning

Sevan G. Ficici and Jordan B. Pollack

DEMO Lab—Department of Computer Science
Brandeis University, Waltham Massachusetts 02454 USA
www.demo.cs.brandeis.edu

**Abstract.** We develop a novel coevolutionary algorithm based upon the concept of Pareto optimality. The Pareto criterion is core to conventional multi-objective optimization (MOO) algorithms. We can think of agents in a coevolutionary system as performing MOO, as well: An agent interacts with many other agents, each of which can be regarded as an objective for optimization. We adapt the Pareto concept to allow agents to *follow gradient* and *create gradient* for others to follow, such that coevolutionary learning succeeds. We demonstrate our Pareto coevolution methodology with the *majority function*, a density classification task for cellular automata.

## 1 Introduction

The challenge facing an agent situated in a coevolutionary domain can be described as a form of multi-objective optimization (MOO) where every other agent encountered constitutes an objective to be optimized. That the techniques of conventional MOO, most notably those incorporating the notion of *Pareto optimality*, may be usefully applied to coevolution has recently been suggested [4, 20], but has yet to be deeply explored. This paper begins our investigation into the connection between coevolution and multi-objective optimization, paying particular attention to how the Pareto optimality concept may lead to methods that address issues unique to coevolution. (See Noble and Watson [13] for another example of a "first-generation" Pareto coevolutionary algorithm.)

All coevolutionary systems involve concurrent processes of gradient creation and gradient following. The central themes of coevolution research concern the ways in which these processes interact to dynamically modify the scope of interactions that occur between coevolving entities. Hillis' [6] seminal work on coevolving a population of sorting networks against a population of input vectors gives a compelling illustration of how these processes may interact. Hillis recognizes that the feedback loops between the processes of gradient creation and gradient following can behave like an optimizer with a dynamically adjustable evaluation function; by judiciously selecting the range of test cases applied to the sorting networks, coevolution can make more effective use of finite computational resources to achieve better results.

Nevertheless, the ability to dynamically alter a learning landscape does not by itself guarantee that coevolution will lead to effective learning. Indeed, conventional coevolutionary methods are known frequently to exhibit a number of irksome modes of behavior that hinder learning. Intransitive superiority relationships [3] and mediocre-stable states [17] are two examples.

Our experiments involve the coevolution between cellular automata (CA) rules for a density classification task (the *majority function*) and initial condition densities. We choose this problem because an extensive literature on the majority function exists (e.g., [14, 21, 1, 11, 12]) and the best rules currently known derive from coevolution (by Juillé and Pollack, with success rates of 85.1%, 86.0% [10], and 86.3% [9]), allowing us to more meaningfully ascertain the significance of our results. We are able to discover rules with a competitive success rate of 84.0%. While not superior to the results of Juillé and Pollack, our rule is significantly better than all results published elsewhere, and we anticipate improved results, as we discuss below. More importantly, we argue that our method offers a potentially more general and comprehensive approach towards coevolution.

This paper is organized as follows: Section 2 details how we employ the Pareto optimality criterion in our coevolutionary algorithm; Section 3 describes the majority function and how we deploy our Pareto coevolutionary algorithm to work on it; Section 4 gives additional details of our experiment setup; Section 5 reviews results; Section 6 discusses plans for new experiments and concludes.

## 2 Pareto Coevolution Methodology

### 2.1 Learning from Teaching

We equate the processes of gradient following and gradient creation, discussed above, with the roles of *learning* and *teaching*, respectively. Every agent with which a learner interacts is a teacher, every agent with which a teacher interacts is a learner. An agent in a coevolutionary framework usually, though not always, plays both roles simultaneously; and, when both roles are played by an agent, they are usually not performed with equal success. Our approach to coevolution considers the roles of learning and teaching to be orthogonal.

The only evidence that a learner has to indicate success at following gradient comes from achieving good outcomes (high payoffs) through interaction with agents (teachers). But, the learning gradient is typically of very high dimension in coevolution, since each agent with which a learner can interact represents a dimension to be optimized. Therefore, some principled method of integrating this space is required, and this is the reason we look towards the Pareto optimality concept for help.

We define the role of the teacher to be one that is awarded fitness for providing gradient for learners. But, what evidence can we gather to infer success at this job? We define the *learnability* of a teacher, with respect to a particular learner, to be the likelihood that the learner can be transformed, over some number of variation steps, to become competent (or more competent) at the task posed by the teacher. (Note that learnability becomes sensitive to the state of an entire population if recombinative methods are used in variation.) Thus, the task posed by a teacher is unlikely to be learned if the learner is too remote from regions in variation space where learners are competent at the task—the teacher is “too difficult.” Teachers that are completely mastered by a learner also have a low learnability in the sense that the learner has no chance of improving its performance on the task, since it is already perfect.

Rather than try to measure teacher learnability directly, our approach is to discover teachers that can demonstrate *gaps* in learner competence—i.e., show one learner to be more proficient than another at a particular task. We operate on the intuition that teachers that fill competence gaps are likely to be learnable because they expose and explore pre-existing gradients of learner ability in different dimensions of behavior. We rely on the variation process to open new dimensions. This way, we can hope always to have relevant challenges that are of appropriate difficulty. Note that, if an evolving population contains a learner (call it $L^*$) that is superior to all other learners in the population with respect to every teacher, then no competence gaps exist “above” $L^*$ to fill with teachers that are certain to challenge it. We must wait until some variation occurs to generate a new learner that outperforms $L^*$ in some dimension(s) and creates new gaps in competence. This approach to creating and maintaining gradient for coevolutionary learning is substantially different from those of Rosin [18], Juillé [7], Olsson [15], and Paredis [16].

## 2.2 Learning: Following Gradient

This section describes how we measure success at following a high-dimensional gradient using the Pareto optimality concept. We name the set of learners $\mathcal{R}$ and the set of teachers $\mathcal{S}$. The payoff matrix $\mathbf{G}$ describes the performance of all learners against all teachers, where $\mathbf{G}_{i,j}$ is the payoff earned by learner $i$ when interacting with teacher $j$.

- Learner $x$ *Pareto dominates* learner $y$ with respect to the set of teachers $\mathcal{S}$, denoted as $x \overset{\mathcal{S}}{\succ} y$, iff: $\forall w \in \mathcal{S} : \mathbf{G}_{x,w} \geq \mathbf{G}_{y,w} \quad \wedge \quad \exists v \in \mathcal{S} : \mathbf{G}_{x,v} > \mathbf{G}_{y,v}$.
- Learners $x$ and $y$ are *mutually non-dominating*, denoted as $x \overset{\mathcal{S}}{\simeq} y$, iff: $\exists w, v \in \mathcal{S} : \mathbf{G}_{x,w} > \mathbf{G}_{y,w} \quad \wedge \quad \mathbf{G}_{x,v} < \mathbf{G}_{y,v}$.
- The *Pareto front* of a set of learners $\mathcal{R}$, denoted as $F^0(\mathcal{R})$, is the subset of all non-dominated learners in $\mathcal{R}$: $F^0(\mathcal{R}) = \{x \in \mathcal{R} : \nexists w \in \mathcal{R},\ w \overset{\mathcal{S}}{\succ} x\}$.
- The *dominated subset* of $\mathcal{R}$, denoted as $D^0(\mathcal{R})$, is the subset of learners that are dominated by some learner in $\mathcal{R}$: $D^0(\mathcal{R}) = \{x \in \mathcal{R} : \exists w \in \mathcal{R},\ w \overset{\mathcal{S}}{\succ} x\}$.
- Note that a learner belongs exclusively either to the front or the dominated set: $F^0(\mathcal{R}) \cap D^0(\mathcal{R}) = \emptyset$ and $F^0(\mathcal{R}) \cup D^0(\mathcal{R}) = \mathcal{R}$.

Once we identify the Pareto front $F^0$ and the set of dominated learners $D^0$, we may compute the next *Pareto layer*, $F^1 = F^0(D^0(\mathcal{R}))$—the set of learners that are non-dominated once we exclude the Pareto front. We may continue this process until every learner is understood to belong to a particular Pareto layer. Pareto layers indicate both generality and uniqueness in learner competence: Every learner in $F^n$ is less broad in competence than some learner in $F^{n-1}$, and every learner in $F^n$ can do something better than some other learner in $F^n$.

**Intra-Layer Ranking** High-dimensional spaces of competing objectives are known to cause problems for Pareto ranking in ordinary (non-coevolutionary) EAs [5]. As we note above, a large number of teachers gives a high-dimensional gradient. This creates potentially many ways in which a learner may excel and earn a place in a particular Pareto layer. Indeed, we find in our experiments that the number of learners in $F^0$ tends to increase over evolutionary time and may ultimately include as much as 75% of the entire population.

We therefore require some form of *intra-layer ranking* to differentiate learners within a particular (possibly crowded) layer. We consider two approaches, both stemming from diversity-maintenance techniques, and find them to behave similarly (the experiments reported in this paper use the second approach). Our first approach is similar to Juillé and Pollack's [8] *competitive fitness paradigm*. In comparing two learners from the same layer, we give each learner a point for each dimension (teacher) in which it out-scores the other learner. (Alternatively, if learners $x$ and $y$ out-score each other in $n_x$ and $n_y$ dimensions, respectively, then the better of the two agents gets $|n_x - n_y|$ points and the other gets 0 points.) We accumulate points over all pair-wise comparisons of learners *that belong to the same layer*, and then rank learners according to the sums. Our second approach applies Rosin's [18] *competitive fitness sharing* method within each layer. The learners within a layer are then ranked with respect to each other according to the results of the fitness sharing. Regardless of which intra-layer ranking approach is used, *inter*-layer ranking is achieved by giving the highest-ranked learner(s) of Pareto layer $F^n$ a global rank just below the lowest-ranked learner(s) of Pareto layer $F^{n-1}$.

### 2.3 Teaching: Creating Gradient

This section describes how we measure success at creating gradient for learning. We begin with the $m$ by $n$ payoff matrix $\mathbf{G}$, where $m$ is the number of learners and $n$ is the number of teachers. Matrix entry $\mathbf{G}_{i,j}$ is the payoff received by learner $i$ when it interacts with teacher $j$. If learner $x$ performs better than learner $y$ with respect to teacher $j$ (i.e., $\mathbf{G}_{x,j} > \mathbf{G}_{y,j}$), denoted $x \overset{j}{>} y$, then we say that teacher $j$ *distinguishes* the learner pair $(x, y)$ in favor of $x$. By causing a pair of learners to receive different payoffs, a teacher exposes a gap in the proficiencies of the two learners. We are interested to identify all such proficiency gaps in the population of learners, as made apparent by the population of teachers.

To identify all learner pairs that are distinguished by each teacher, we construct a new $n$ by $m^2 - m$ matrix $\mathbf{M}$. Each column of this matrix corresponds to a particular pair-wise comparison of learners across all $n$ teachers. We exclude self-comparisons, since there cannot exist any proficiency gaps between a learner and itself, and we treat the learner pairs (A, B) and (B, A) as distinct—(A, B) is reserved for teachers that distinguish A and B in favor of A, while (B, A) is for teachers that distinguish in favor of B. The matrix entry $\mathbf{M}_{j,k}$ equals one if teacher $j$ distinguishes the learners in pair $k = (x, y)$ in favor of $x$. Clearly, if entry $\mathbf{M}_{j,k}$ is non-zero, then entry $\mathbf{M}_{j,k'}$ must be zero, where $k = (x, y)$ and $k' = (y, x)$.

$$\mathbf{G} = \begin{array}{c|ccc} & \alpha & \beta & \gamma \\ \hline \mathrm{A} & 1 & 1 & 3 \\ \mathrm{B} & 2 & 3 & 2 \\ \mathrm{C} & 1 & 2 & 1 \end{array} \qquad \mathbf{M} = \begin{array}{c|cccccc} & (\mathrm{A},\mathrm{B}) & (\mathrm{A},\mathrm{C}) & (\mathrm{B},\mathrm{A}) & (\mathrm{B},\mathrm{C}) & (\mathrm{C},\mathrm{A}) & (\mathrm{C},\mathrm{B}) \\ \hline \alpha & 0 & 0 & 1 & 1 & 0 & 0 \\ \beta & 0 & 0 & 1 & 1 & 1 & 0 \\ \gamma & 1 & 1 & 0 & 1 & 0 & 0 \end{array} \tag{1}$$

Once we obtain matrix **M**, we have identified all the ways in which each teacher demonstrates utility as a touchstone of learner competence. But, how shall we use this information to create a selective pressure for teachers? To begin with, a teacher that fails to reveal any variation in learner ability is dubious.

In matrix **M** (Equation 1), we see that teacher $\beta$ not only distinguishes all the learner pairs that teacher $\alpha$ does, namely pairs (B, A) and (B, C), but also distinguishes another pair, (C, A). If we were to apply Pareto ranking to matrix **M**, then we would conclude that teacher $\beta$ is superior to $\alpha$ because its ability to reveal variations in learner competence is in some sense more general. Let us take this example to an extreme and imagine teacher $\alpha$ to distinguish only a single pair $(a, b)$ while $\beta$ distinguishes $(a, b)$ and very many more. What these two dramatically different teacher profiles tell us is that the kind of challenge offered by teacher $\alpha$ must be very unlike that offered by $\beta$. Therefore, even though $\beta$ supposedly dominates $\alpha$, teacher $\alpha$ reveals a dimension of variation in the learner pair $(a, b)$ that $\beta$ does not. For this reason, Pareto ranking of matrix **M** is inappropriate.

We may instead give all teachers that distinguish at least one learner pair a score of one, and then divide each teacher's score by the number teachers with an identical profile (to better maintain diversity). But, we feel this method to be too coarse-grained. Our compromise approach is to perform fitness sharing much like Rosin [18].

Equation 2 shows that the score received by teacher $j$ is the sum, across all learner pairs distinguished by it, of the *value* of a learner pair divided by the pair's *discount factor*. The discount factor of a pair is the total number of teachers that distinguish it. For the value of a pair, we experiment with two possibilities. Our first approach (Equation 3, left) simply gives every pair an equal value ($v_k = 1$). Our second approach (Equation 3, right) recognizes certain pairs as more significant than others. For example, we may argue that a teacher should get more reward for distinguishing between two good learners than for distinguishing between two poor ones. Further, a teacher should be rewarded more for showing a generally good learner to do something worse than a generally poor learner than the other way around. Therefore, our second approach assigns the value of a learner pair $k = (x, y)$ to be the fitness of the loser $y$.

$$s_j = \sum_k \mathbf{M}_{j,k} \frac{v_k}{d_k} \qquad d_k = \sum_i \mathbf{M}_{i,k} \tag{2}$$

$$v_k = 1 \quad \textbf{OR} \quad v_k = \text{fitness}(y), \text{where } k = (x, y) \tag{3}$$

## 3 Majority Function

### 3.1 Description

Density classification tasks are a popular area of study in cellular automata research [14, 21, 9, 10, 1, 11, 12]. The objective is to construct a rule that will cause a one-dimensional, binary CA to converge, within some pre-determined number of time-steps, to a state of all ones if the percentage (or *density*) of ones in the initial condition (IC) is greater than or equal to some pre-established value, $\rho \in [0, 1]$. Otherwise, the rule should cause the CA to converge to a state of all zeros. The majority function uses a value of $\rho = 0.5$.

Though Land and Belew [11] prove that no rule correctly classifies all initial conditions, the highest possible success rate remains unknown. Currently, the best rules (of radius three, operating on a CA lattice of 149 bits) achieve success rates of 85.1%, 86.0%, and 86.3% over a uniform sampling of initial conditions, and were discovered by Juillé and Pollack [10, 9] through coevolution. These rules represent a significant improvement over all earlier rules, for example ABK (82.4%), DAS (82.3%), and GKL (81.5%), as discussed in [10, 14].

### 3.2 Coevolution of Rules and Densities

Following Juillé and Pollack [10, 9], we use two-population coevolution to find CA rules of radius three that operate on a one-dimensional lattice of 149 bits. A radius of $n$ means that lattice positions $i - n$ through $i + n$ (with wrap-around) are used by the CA rule to determine the next state of lattice position $i$. A radius of three gives a "window" of seven bits, meaning the rule must contain $2^7 = 128$ bits, one for each possible window state. This gives us a search space of $2^{128}$ possible rules. A lattice of 149 bits has $2^{149}$ possible states, and therefore initial conditions. While one population evolves rules (bit-strings of length 128), the other population evolves initial condition *densities* (floating-point numbers)—not actual initial conditions.

Our experiments depart from those of Juillé and Pollack by using our Pareto coevolution methodology, outlined above. As with many two-population coevolutionary domains, ours has an intrinsic asymmetry in the difficulties faced by the two populations: the discovery of good rules is much harder than the discovery of challenging IC densities. Though the space of possible initial conditions is much larger than the space of possible rules, there exist only 150 distinct IC densities (from all-zeros to all-ones). Further, densities generally become more difficult as they approach 0.5 [12, 11], so the space in some sense approximates a uni-modal landscape. For these reasons, rules are assigned only the role of learners and IC densities are assigned only the role of teachers. We can imagine other coevolutionary domains where coevolving entities would be called upon to fulfill both roles, such as Tic-Tac-Toe.

### 3.3 Derivation of Payoff Matrix: Test Case Sampling

All ranking methods potentially impose severe non-linearities by expanding small differences in performance and compressing large ones. Pareto ranking is no exception and may even be considered more extreme in some ways. Because of this non-linear behavior, the payoff matrix $\mathbf{G}$, which forms the basis of our approach, should be as accurate as possible. Thus, our approach is at least as sensitive to non-deterministic domains as conventional coevolutionary methods.

Our CA domain, however, is deterministic. Because we evolve rules against densities rather than actual initial conditions, rule performance against a particular density is expressed as the percentage of correctly classified ICs of that density class. But, we generally cannot afford to sample a particular density class exhaustively. This impedes our ability to compare rules of similar ability. Further, as rules improve in performance, we generally rely on densities closer to 0.5 to distinguish them. Yet, the distribution of initial conditions over densities is binomial, which makes our sampling of ICs significantly more sparse as densities approach 0.5 (and our estimation of rule performance less accurate).

We desire a method to extract meaningful information about rule performance with relatively few samples. Fortunately, for our Pareto coevolution methodology to work, we do not need to know precisely how well a particular rule (learner) performs against a particular IC density (teacher); we only need relative ranking information. Our solution is to once again turn to the Pareto optimality criterion. Note that our use of Pareto ranking to derive the payoff matrix (described here) is not to be confused with our use of Pareto ranking to rate learners once payoffs are known (described in Section 2.2).

We compute each column $j$ of the payoff matrix $\mathbf{G}$ in the following manner. We generate some number $q$ of initial conditions ($q = 40$ in our experiments) that are representative of density (teacher) $j$; all $m$ rules (learners) are tested on this set of ICs. The test results are placed in an $m$ by $q$ matrix $\mathbf{H}$, where $\mathbf{H}_{u,v} = 1$ if rule $u$ correctly classifies IC $v$ and $\mathbf{H}_{u,v} = 0$ otherwise. We then perform Pareto ranking of rule performance based on $\mathbf{H}$. Rules on the Pareto front $F^0$ are given the highest payoff, those in layer $F^1$ the next highest, and so on. These payoffs form column $j$ of the payoff matrix $\mathbf{G}$.

All rules cover some subset of ICs that belong to a particular density class. These subsets may overlap in any number of ways. We require that a rule dominate another in terms of *measured coverage* in order to receive a higher payoff; this is more stringent than mere comparison of *measured success rates*. For example, two rules that each cover 50% of some density class may overlap entirely on the one extreme, or not at all on the other extreme. Regardless of the amount of actual overlap, the chance that our measured coverage will indicate one rule to dominate the other is extremely small, even though the measured success rates (number of ICs solved by the two rules) will very likely be unequal. But, as the actual performance rates of two rules grow apart, the more likely it becomes for the better of the two to dominate the other in measured coverage (especially if actual coverage of the stronger rule overlaps heavily with that of the other).

Our method of test-case sampling removes the need for an explicit similarity metric—similarity is guaranteed by the process of generating multiple ICs from a density value. Juillé and Pollack [10, 9] require a similarity metric to cluster IC densities so that rule performance can be gauged with respect to density.

## 4 Experimental Setup

A rule is given 320 time-steps to converge the CA to the correct state. The sizes of our populations of rules and IC densities are $N_R = 150$ and $N_{IC} = 100$, respectively. All rules are tested against all densities in every generation. Each density is sampled 40 times, giving a total of $6 \times 10^5$ evaluations per generation.

The initial population of rules is composed of random bit-strings, distributed uniformly over the range of string densities (0 to 128 ones). The initial population of IC densities is distributed uniformly over the interval [0, 1]. Rules are varied by one-point crossover and a 2% per-bit mutation rate. IC densities are varied by the addition of Gaussian noise of zero mean and standard deviation of 0.05.

Rule ranks are squared before they are normalized for the roulette wheel. The next generation of rules is created by using Baker's [2] SUS method to select 75 rules that remain unaltered and another 75 rules to which the variation operators are applied. The next generation of IC densities is created by using SUS to select 50 densities that remain unaltered and another 25 that are varied. The remaining 25 densities are picked at random from a uniform distribution.

An IC density is converted to an actual initial condition by first adding Gaussian noise of zero mean and standard deviation of 0.05. We multiply the result by 149 and take the floor to arrive at an integer between 0 and 149. The initial condition will be a random bit-string of exactly that number of ones. All rules see the same set of initial conditions.

## 5 Results and Discussion

We have conducted six runs of our experiment, three for each of our two methods of valuating learner pairs (see Section 2.3 and Equation 3). Our best result to-date is a rule that correctly classifies 84.0% of initial conditions, shown in Table 1. We determine this success rate by testing the rule against $4 \times 10^7$ randomly generated ICs (that is, a binomial distribution of IC densities). The rule took approximately 1300 generations to evolve. Two of the other runs each exceed 81% success; our worst result is 78.8% success. While our best result comes from our first method of valuating learner pairs (see Equation 3), the data do not distinguish the performance of the two methods.

In experiments where $N_R = N_{IC} = 400$ [10], Juillé and Pollack discover a rule that achieves 85.1% success (some runs give $\leq 76\%$). Their best results (86.0% and 86.3% success) are discovered in experiments where $N_R = N_{IC} = 1000$ [10, 9] (experiments last 5000 generations; all exceed 82.0% success). They test all rules against all densities, but each density is sampled only once. In contrast, our experiments use much smaller population sizes ($N_R = 150, N_{IC} = 100$),

**Table 1.** Currently best evolved rule using Pareto coevolution.

| | |
|---|---|
| Rule 1 | 00010000 01010011 00000000 11010010 00000000 01010001 00001111 01011011 |
| 84.0% | 00011111 01010011 11111111 11011111 00001111 01010101 11001111 01011111 |

but we sample each density 40 times. Thus, the total amount of computation in our experiments falls in between those of Juillé and Pollack. But, in exchange for a more expensive IC density sampling procedure (see Section 3.3), we avoid the explicit similarity metric required by Juillé and Pollack to classify ICs, and thereby arrive at an approach that should more easily generalize to other problem domains (e.g., sorting networks). Indeed, we intend ultimately to apply our Pareto coevolution methodology to variable-sum games, in addition to zero-sum games such as those studied by Rosin [18] and Juillé [7].

While we do not improve upon the results of Juillé and Pollack [10, 9], we improve significantly upon all results published elsewhere. The next most effective rule is by Andre, et al [1], which performs at 82.4%. This rule was discovered with genetic programming in experiments using a rule population size of 51,200, each tested on 1000 different ICs (ICs were not coevolved) per generation. We are confident that we can improve our results with larger populations.

## 6 Conclusion and Future Work

We propose a novel coevolutionary algorithm based upon the concept of Pareto optimality. Our algorithm distinguishes the role of the gradient follower from that of the gradient creator, even though both may coexist within the same agent, and utilizes Pareto-inspired metrics of success for both roles. We use our algorithm to coevolve cellular automata rules for the majority function and discover a rule that correctly classifies 84.0% of initial conditions. Though our result is encouraging, we clearly have many more experiments to perform. We must try larger populations, and perhaps different inter-generational replacement schemes. We require control experiments to identify the contributions of each component of our overall methodology. In these controls, we will substitute one or two of our methods (for rewarding learners and teachers, and for sampling densities) with more conventional mechanisms, for example using a problem-specific similarity metric instead of our more expensive sampling method. We are conducting a deeper analysis of our algorithm's dynamics. Particularly, a more game-theoretic review of our algorithm is necessary to fully expose its behavior in variable-sum games and zero-sum games with intransitive superiority relationships. Finally, we are investigating various ways to integrate our metrics of learner and teacher success for single-population coevolution.

## Acknowledgments

The authors thank Anthony Bucci, Hugues Juillé, Norayr Vardanyan, Richard Watson, and members of the DEMO Lab.

## References

1. D. Andre et al. Evolution of intricate long-distance communication signals in cellular automata using genetic programming. In C. G. Langton and K. Shimohara, editors, *Artificial Life V*, pages 16–18. MIT Press, 1996.
2. J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proc. Second Int. Conf. on Genetic Algorithms*, pages 14–21. Lawrence Earlbaum, 1987.
3. D. Cliff and G. F. Miller. Tracking the red queen: Measurements of adaptive progress in co-evolutionary simulations. In F. Moran et al., editors, *Third Euro. Conf. on Artificial Life*, pages 200–218. Springer, 1995.
4. S. G. Ficici and J. B. Pollack. A game-theoretic approach to the simple coevolutionary algorithm. In Schoenauer et al. [19], pages 467–476.
5. C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3(1):1–16, 1995.
6. D. Hillis. Co-evolving parasites improves simulated evolution as an optimization procedure. In C. Langton et al., editors, *Artificial Life II*, pages 313–324. Addison-Wesley, 1991.
7. H. Juillé. *Methods for Statistical Inference: Extending the Evolutionary Computation Paradigm*. PhD thesis, Brandeis University, 1999.
8. H. Juillé and J. B. Pollack. Co-evolving intertwined spirals. In L. J. Fogel et al., editors, *Proc. Fifth Annual Conf. on Evolutionary Programming*, pages 461–468. MIT Press, 1996.
9. H. Juillé and J. B. Pollack. Coevolutionary learning: a case study. In J. Shavlik, editor, *Proc. Fifteenth Int. Conf. on Machine Learning*, pages 251–259. Morgan Kaufmann, 1998.
10. H. Juillé and J. B. Pollack. Coevolving the "ideal" trainer: Application to the discovery of cellular automata rules. In J. R. Koza et al., editors, *Proc. Third Annual Conf. on Genetic Programming*, pages 519–527. Morgan Kaufmann, 1998.
11. M. Land and R. K. Belew. No perfect two-state cellular automata for density classification exists. *Physical Review Letters*, 74(25):1548–1550, 1995.
12. M. Mitchell et al. Evolving cellular automata to perform computations: Mechanisms and impediments. *Physica D*, 75:361–391, 1994.
13. J. Noble and R. A. Watson. Pareto coevolution: Using performance against coevolved opponents in a game as dimensions for pareto selection. In L. Spector et al., editors, *Proc. 2001 Genetic and Evo. Comp. Conf.* Morgan Kaufmann, 2001.
14. G. M. B. Oliveira et al. Evolving solutions of the density classification task in 1d cellular automata, guided by parameters that estimate their dynamic behaviour. In M. A. Bedau et al., editors, *Artificial Life VII*, pages 428–436. MIT Press, 2000.
15. B. Olsson. *NK*-landscapes as test functions for evaluation of host-parasite algorithms. In Schoenauer et al. [19], pages 487–496.
16. J. Paredis. Towards balanced coevolution. In Schoenauer et al. [19], pages 497–506.
17. J. B. Pollack and A. D. Blair. Co-evolution in the successful learning of backgammon strategy. *Machine Learning*, 32(3):225–240, 1998.
18. C. D. Rosin. *Coevolutionary Search Among Adversaries.* PhD thesis, University of California, San Diego, 1997.
19. M. Schoenauer et al., editors. *Parallel Prob. Solv. from Nature 6.* Springer, 2000.
20. R. A. Watson and J. B. Pollack. Symbiotic combination as an alternative to sexual recombination in genetic algorithms. In Schoenauer et al. [19], pages 425–434.
21. J. Werfel et al. Resource sharing and coevolution in evolving cellular automata. *IEEE Transactions on Evolutionary Computation*, 4(4):388–393, 2000.