Robustness in Cooperative Coevolution

R. Paul Wiegand
US Naval Research Laboratory
4555 Overlook Ave, SW
Washington, DC USA
paul@tesseract.org

Mitchell A. Potter
US Naval Research Laboratory
4555 Overlook Ave, SW
Washington, DC USA
mpotter@aic.nrl.navy.mil

ABSTRACT

Though recent analysis of traditional cooperative coevolutionary algorithms (CCEAs) casts doubt on their suitability for static optimization tasks, our experience is that the algorithms perform quite well in multiagent learning settings. This is due in part because many CCEAs may be quite suitable to finding behaviors for team members that result in good (though not necessarily optimal) performance but which are also robust to changes in other team members. Given this, there are two main goals of this paper. First, we describe a general framework for clearly defining robustness, offering a specific definition for our studies. Second, we examine the hypothesis that CCEAs exploit this robustness property during their search. We use an existing theoretical model to gain intuition about the kind of problem properties that attract populations in the system, then provide a simple empirical study justifying this intuition in a practical setting. The results are the first steps toward a constructive view of CCEAs as optimizers of robustness.

Categories and Subject Descriptors

I.2.m [**Artificial Intelligence**]: Miscellaneous

General Terms

Algorithms, Theory, Experimentation

Keywords

Robustness, coevolutionary algorithms, cooperative coevolution, compositional systems

1. INTRODUCTION

Coevolutionary algorithms (CEAs) are increasingly popular extensions of traditional evolutionary algorithms (EAs). The most fundamental differences between CEAs and EAs stem from the adaptive nature of fitness evaluation in coevolutionary systems: individuals are assigned fitness values based on direct interactions with other individuals. Such systems have historically been categorized as competitive or cooperative. Our interest is in this latter category of cooperative coevolutionary algorithms, which are capable of exploiting the *compositional* nature of learning problems involving multiple, interacting agents. Specifically, compositional approaches use an individual to represent a component of a larger, more structured problem. Such a component receives fitness based on how well it performs in conjunction with individuals from other populations.

In the case of compositional systems, one hopes to establish a parallel adaptive gradient (akin to the notion of an *arms race* [3]), where steady progress is made by mutual and reciprocal adaptations between collaborating groups of individuals. Unfortunately, in spite of their appeal, coevolutionary algorithms are often challenged by seemingly simple problems. Cooperative coevolution has been well-studied over the last decade, both empirically and analytically, and most of what has been learned concerns their application to static optimization problems. Theoretical analysis has shown that traditional versions of these algorithms may not be well-suited for static optimization tasks, and several suggestions for how to modify them for these domains have been proposed [14, 2]. In spite of this, experience in practice has been quite different in other types of problem domains: Cooperative coevolutionary algorithms (CCEAs) seem to produce very good behaviors of multiagent teams [18].

Though there are many reasons why this is the case, one possible explanation involves a change in viewpoint. When we apply our algorithms to multiagent learning problems, we are not necessarily interested in finding the optimal team. Instead, we are interested in finding a team that performs well, but is also *robust* to deviations in individual member behaviors. This form of robustness is desirable because in practice the capabilities of team members, and even the makeup of the team itself, often change over time. Such a viewpoint is consistent with the above literature, which suggests that CCEAs tend to discover individuals that partner well with a broad range of individuals from the other populations [23]. Indeed, our experience suggests that CCEAs are *particularly* well-suited to finding solutions that include a type of robustness of a component's performance with respect to other components with which it interacts.

This paper offers a study of a simple compositional system, taking the first steps toward the constructive view that many CCEAs optimize for some form of robustness. There are two main objectives of the work. First, recognizing that there are many ways to define the term "robustness", and inspired by the *robustness analysis* field of the operations research community, we offer a general framework for defining robustness, including a specific and practical meaning appropriate for our context. Second, we leverage intuition obtained from considering existing formal models of a common CCEA to construct an empirical study that demonstrates that these algorithms are well-suited for finding solutions with this kind of robustness. In general, we believe the paper provides a constructive clarification of the kinds of compositional problems for which many CCEAs are most appropriate.

2. BACKGROUND

2.1 Cooperative Coevolution

Historically, the first application of coevolution to a compositional problem can be traced back to work by Husbands and Mill[7]. This algorithm provided a direct decomposition of a planning and scheduling problem space into components that reflected various shared resources in a machine shop, and it applied parallel EAs to try to find an optimal schedule for those resources. Two very different general frameworks for applying coevolution to compositional problems were provided by Potter [15] and Moriarty and Miikkulainen [12]. This paper focusses on the Potter model, which is a more generalized form of the approach by Husbands and Mill. Potter defines a multipopulation architecture for cooperative coevolution to address compositional problems. Here each population contains individuals that represent a particular component of the problem, so that one member from each population is needed in order to assemble a complete solution. Evaluation of an individual from a particular population is performed by assembling the individual with collaborating partners from other populations. Aside from evaluation, the populations are evolved independently.

An example may help clarify things. Suppose we are optimizing a two argument function, $f(x, y)$. One might assign individuals in the first population to represent the x argument and individuals in the second population to represent y. Each population is evolved separately, except that when evaluating an individual in some population (e.g., x), collaborating representatives must be chosen from the other population (y) in order to obtain an objective function value with a complete solution, $f(x, y)$. A simple example collaboration method is to choose each of the members of the other population, collaborate to compute an objective value for each interaction, then assign the average value of all such interactions to the individual in the first population as fitness. Selecting a single random individual to serve as a collaborator can be seen as a kind of probabilistic approximation of this method.

There have been a variety of applications of this approach. Indeed, cooperative coevolution has proved to be an effective method for many types of learning problems (e.g., learning constructive neural networks [17], and rule learning [16, 19]), including and especially multiagent learning problems [18] where the goal is not always optimality.

Analytical work surrounding CCEAs has shown that the algorithms are not necessarily attracted to the *optimal collaboration*[1] and may not be well-suited for static optimization tasks in its most naïve form [23]. In fact, there has been some work attempting to modify CCEAs to *correct* this problem either by biasing the algorithm toward the optimum [14] or by making use of Pareto selection and memory mechanisms [2]. While these studies concentrate on making cooperative coevolution a better static optimizer, we concentrate our study on beginning to understand why CCEAs appear to perform well in spite of this in multiagent learning settings. We believe coevolution's success in these domains is a result of its ability to find solutions that have certain specific robustness properties. In particular, coevolution is capable of producing teams of agents that perform well even when some subset of the team does something unexpected, and this is preferable to more optimal, yet brittle teams, that cannot admit such changes.

[1] *Optimal collaboration* is defined as the composite team with the optimal performance.

2.2 Robustness Analysis

The evolutionary computation community has tended to avoid rigorously defining "robustness" in part because specifying a particular definition for the word can be troublesome. Is an agent that performs well *on average* in many situations robust? Is an agent that avoids some kind of catastrophic failure when faced with change robust? Is the agent robust only when it *always* performs well in a subset of similar, but somewhat different situations? If the solution is robust to local changes, but not to more global changes, is it still robust in the general sense of the word? In the midst of these many questions, one thing is clear: There are many valid interpretations of the word "robust". What is often *unclear* is which one a particular researcher is meaning to convey.

Indeed, the Santa Fe Institute has recognized the complications surrounding this topic and established a program on robustness [8]. This program provides a loose forum for a shared discussion and bibliography of works studying issues surrounding robustness (e.g., the differences between notions of "stability" of a system and "robustness" of a system, [9]); however, though the program admits the difficulty in defining robustness, its response to this challenge is to collect many definitions rather than to consolidate them.

The operations research community has taken the opposite, comprehensive approach in a field they call *robustness analysis* (RA) [20]. The idea is to produce a *framework* for rigorously defining robustness in many circumstances. RA asserts that when one analyzes robustness, one is typically attempting to justify the conclusions (solutions) of some model (problem) such that it holds for other "reasonable" combinations of parameter values of that solution. One *first* obtains a specific set of parameter values that constitute a solution to a problem, *then* one attempts to see whether or not those parameter values are somehow "central" in the sense that certain changes in the parameter values still result in something acceptable.

Robustness analysis typically partitions the domain space of potential solution into two subspaces. The first subspace is often referred to as the space of *design* parameters, the second *control* parameters. When we talk of robustness, we generally speak of the relationship of parameter values between these two subspaces: A specific design parameter value may or may not be robust over some set of control parameter values, or vice-versa [13]. When we are assessing the robustness of a particular design parameter value by considering the entire control parameter subspace, we use the term *perfectly* [20] or *absolutely* [4] robust. Alternatively, imperfect robustness may employ measures of distance in the domain space, or probability of inclusion in the comparison set. In other words, one might intend the term "robustness" to apply globally across some subspace, or one might intend a more local meaning of the word.

However the set of control values are chosen, a particular combination of control parameter value and the fixed design parameter value contributes to robustness if they obey some kind of *robustness criterion.* There are many such criteria, such as those resulting in "similar" values determined by some distance metric [21], or those that result in the optimal performance in the worst-case pairing [10], or those that obey a kind of game-theoretic min-max regret rule [13]. In general, this criterion is simply a rule to be applied for each comparison; it can be any rule.

Establishing a clear framework, as has been done to some extent in the OR literature, and as we will attempt to do in this paper for coevolutionary computation, allows researchers to be clear what they mean by the term in their situation. Moreover, it permits outside researchers the ability to consider whether or not the proffered definition has any value to their own research.

3. ROBUSTNESS IN COEVOLUTION: A FRAMEWORK

As in the field of RA, we find it useful to concentrate on two key elements of the process of defining robustness. First, the process involves partitioning the search space in such a way that one could discuss the robustness of one set of variables with respect to another. Second, a rule is provided for comparison purposes. Robustness of a value in the first partition is determined by using that rule to compare the value against all, or a subset of all values in the second set.

While this partitioning of the argument space could be done in any way, in a multipopulation coevolutionary setting there is, of course, a very natural way to determine such a partitioning: along the population boundaries. Recall that a solution for a compositional problem is a composite of components assembled from each population. We divide the process of determining robustness into stages, where each component in a particular solution is evaluated for robustness with respect to the other potential components.

For simplicity and exposition, we consider only two-population CCEAs (thus only two components), but it should be clear that the framework can easily be extended to include coevolutionary systems with more than two populations. We start by considering a particular component.

Definition 1. *Let* $A = \{a_1, a_2, \ldots, a_n\}$ *and* $B = \{b_1, b_2, \ldots, b_n\}$ *represent the set of possible component values for the first and second population, respectively.*

We call $\mathcal{C} : A \times B \mapsto \{0, 1\}$ *a* ***robust criterion****. We say that a particular* $\bar{a}$ *is* ***perfectly robust*** *over* B *with respect to the robustness criterion* $\mathcal{C}_a$ *if* $\forall b_j \in B, \mathcal{C}_a(\bar{a}, b_j) = 1$.

We call $\mathcal{D}_a(\bar{a}, B) = \frac{\sum_j \mathcal{C}_a(\bar{a}, b_j)}{|B|}$ *the* ***degree of robustness*** *of* $\bar{a}$ *over* B *with respect to the robustness criterion* $\mathcal{C}_a$.

Though the above definition is from the perspective of the first component, it is easy to see that we can reverse the two components and apply the same logic to get $\mathcal{D}_b(b_j, A)$, the degree of robustness of b_j over A with respect to $\mathcal{C}_b$. When the partitioning and $\mathcal{C}$ are fully specified, these definitions allow a researcher to clearly and precisely define what it means to say that a particular component is more robust than another component: it is more robust if its degree of robustness is higher, i.e., component value a_i is more robust than a_j if $\mathcal{D}_a(a_i, B) > \mathcal{D}_a(a_j, B)$. Of course, in practice one will not have access to the entire space of potential components with which to compare, and some sampling method for estimating the degree of robustness will be required.

Of course, rather than a binary robustness criterion, we might have used a parametric function of some kind. Such a function would have the advantage of retaining information about relative objective value differences; however, here we will remain consistent with the OR literature and consider robustness criteria to be merely indicator functions.

The trick in using this framework is the selection of an appropriate $\mathcal{C}$ function. The robustness criterion may be any condition, but the usefulness of the criterion for the purposes of analysis or comparison is limited to the degree to which the criterion has any real meaning to researchers. Still, a precise instantiation of a robustness definition will help focus what is being studied, and permit other researchers to agree or disagree with the utility of the definition.

3.1 A Practical Partitioning

The above framework assumes a partitioning along population boundaries, but in general the ideas discussed here are not limited to such partitioning. It is only important that the division is made and that the researcher is clear about the variables being compared. However, we restrict our studies to population-based partitions for several reasons. First, since the engineer has already made partitioning decisions surrounding the relationships of the components by their very decomposition in the CCEA representation, it is reasonable to assume that these same representational boundaries reflect good choices for robustness analysis purposes — if for no other reason than they are commensurate with what the engineer thinks is important (rightly or wrongly). Second, given that our problem domains of interest are multiagent learning tasks, it makes sense to decompose such problems along the agent boundaries both for representational and robustness analysis purposes. That is, the robustness we care about is primarily the robustness of team performance to changes in individual team members.

3.2 Global vs. Local Robustness

Here, we consider robustness from a global perspective, eschewing the idea of restricting our consideration to local perturbations only. While a global definition for robustness may seem counterintuitive, it is often the most practical. For one thing, there is generally no real topology connecting a given space of strategies (genotypes) outside of the EA itself. Once we have obtained a solution, the connectivity of the strategy space established by genetic operators is irrelevant. Although we are often interested in a topology that allows us to measure robustness over a set of "reasonable" changes to our strategies — changes we can expect when the solution is deployed — defining what such changes are in terms of representation of behaviors is difficult and beyond the scope of this paper. It suffices to say that topologies implied by these considerations may be radically different than those imposed by effective search operators. In existing mathematical models of the CCEA, on which much of our intuition is based, there are no simple local relationships between strategies, and we will not assume any here.

Table 1: This a simple example of a payoff function for a two-component CCEA. In game-theoretic terms, the solutions (a_1, b_1) and (a_3, b_3) are *both* Nash equilibria.

	b_1	b_2	b_3
a_1	**5**	4	2
a_2	4	3	1
a_3	2	1	**6**

3.3 A Practical Robustness Criterion

Using the above framework, we can define a simple and natural notion of robustness helpful for other studies in our lab. We will concentrate on defining robustness from the perspective of the first component; robustness of the second component follows naturally by symmetry. Given an objective function $f : A \times B \mapsto \mathbb{R}$, we choose the following definition for $\mathcal{C}_a$.

$$\mathcal{C}_a\left(\bar{a}, b_j\right) = \begin{cases} 1 & \text{if } f\left(\bar{a}, b_j\right) \geq \sum_i \frac{f\left(a_i, b_j\right)}{|A|} \\ 0 & \text{otherwise} \end{cases}$$

The following example should help make this idea more tangible. Consider the simple two-player 3×3 payoff shown in Table 1. Here there are components A and B, which have access to the potential values $\{a_1, a_2, a_3\}$ and $\{b_1, b_2, b_3\}$, respectively. There are two pure Nash equilibria, with values 5 and 6, respectively. Considering each B component value, we compute the average across all A values as $\{3.\bar{6}, 2.\bar{6}, 3\}$. From this we can compute the degrees of robustness of a_1 and a_3 as $\frac{2}{3}$ and $\frac{1}{3}$ respectively — and symmetrically we can perform the same computations from the perspective of the B component. So, while (a_3, b_3) obtains the optimal objective function value, (a_1, b_1) has better support from other component values and is in that sense more robust.

The definition we provide above asserts that we believe a particular $\bar{a}$ component value to be robust if it maximizes the number of pairings with B component values that have a better than average performance across all possible A components. We believe this to be a meaningful and potentially quite useful definition for robustness for many settings, multiagent learning in particular, because it reflects the ability of the $\bar{a}$ component to perform reasonably well in an environment in which the B component with which it is paired may change over time.

4. EMPIRICAL ANALYSIS

4.1 Max of Two Quadratics Problem Domain

To help illustrate the relationship between CCEAs and our notion of robustness, we use the class of problem domains known as MaxTwoQuadratics (MTQ) [14, 23]. The MTQ domain can be seen as a kind of problem generator capable of producing anything from very simple to very challenging problem instances. At a high level, the traditional MTQ can be simply described as the maximum of two, two-dimensional functions, $\max\{f_1, f_2\}$, where f_1 and f_2 are two quadratic polynomials. The relative positions, heights, and widths of the peaks are parameterized, and particular instances of the problem can be produced by specifying these parameters. We label the peaks 1 and 2 and specify the heights and widths of the peaks with the parameters H_1, H_2, S_1, and S_2, respectively. The position of the first peak is located at $(\bar{x}_1, \bar{y}_1)$, the second at $(\bar{x}_2, \bar{y}_2)$.

$$f_1(x,y) := H_1 \cdot \left[1 - \frac{16}{S_1}\left(x_i - \bar{x}_1\right)^2 - \frac{16}{S_1}\left(y_i - \bar{y}_1\right)^2\right] + K$$
$$f_2(x,y) := H_2 \cdot \left[1 - \frac{16}{S_2}\left(x_i - \bar{x}_2\right)^2 - \frac{16}{S_2}\left(y_i - \bar{y}_2\right)^2\right] + K$$

$$MTQ(x,y) := \max\{f_1, f_2, 0\} \qquad (1)$$

MaxTwoQuadratics can, of course, be viewed simply as an optimization problem; however, if we consider each dimension of the problem as representing strategies for different agents on a team with 2 members, MTQ may also be seen as defining a game payoff function for an abstract multi-agent problem. By viewing it in this way, we can begin to explore the degree to which traditional CCEAs are capable of finding robust solutions to similar types of problems.

In this paper, we are concerned mainly with the relative widths of the peaks, so we position the peaks at $\left(\frac{1}{4}, \frac{1}{4}\right)$ and $\left(\frac{3}{4}, \frac{3}{4}\right)$, with heights $H_1 = 50$ and $H_2 = 150$, $K = 250$. The idea is to produce two peaks at diagonal corners of the unit rectangle, a suboptimal peak (f_1) and an optimal peak (f_2). In game-theoretic terms, this produces a problem with two pure Nash equilibria, or two potential solutions to the problem. By controlling the relative widths of these two peaks (s_1 and s_2), we will be able to indirectly affect the degree of robustness of these two key potential solution sets.

4.2 Analytical Intuition

An appealing abstract mathematical model for cooperative coevolution can be drawn from the biology literature: Evolutionary Game Theory (EGT) [11, 6]. EGT provides a formalism based on traditional game theory and dynamical systems techniques to analyze the limiting behaviors of interacting populations under long-term evolution. For specifics about applying EGT to the analysis of CEAs, see [5, 23]. In this paper, we will use this model only to provide some intuitional validation that simple, idealized CCEAs exploit the above robustness property. We begin with a brief description of the mathematical model — readers interested in more details should consult the available literature.

In a two-population EGT model, a common way of expressing the rewards from individual interactions is through a pair of *payoff matrices*. As it turns out, the particular class of CEAs in which we are interested have a certain kind of symmetric property in the payoff in the sense that when individuals from the first population interact with individuals from the second, one payoff matrix A is used, while individuals from the second population receive rewards defined by the transpose of this matrix (A^T). In our theoretical exploration of EGT in this paper, we will use an *infinite population*. Thus, a population can be thought of not as a set of individuals, but as a finite-length vector $\vec{x}$ of proportions, where each element in the vector is the proportion of a given individual genotype (or strategy) in the population. The proportions in a valid vector must sum to one, and all legal vectors make up what is commonly known as the *unit simplex*, denoted Δ^n, where n here is the number of distinct genotypes possible in the first population, $\vec{x} \in \Delta^n : x_i \in [0,1], \sum_{i=1}^{n} x_i = 1$. In a two-population model, the domain space of the system is a Cartesian product of two such simplexes, $\Delta^n \times \Delta^m$, where m is the number of distinct genotypes possible in the second population. Here we assume, without loss of generality, that $n = m$.

Formally we can model the effects of evaluation and proportional selection over time using a pair of difference equations, one for each population. The proportion vectors for the two populations are $\vec{x}$ and $\vec{y}$ respectively. Neglecting the issue of mutation and breeding, concentrating only on the effects of selection, we can define the dynamical system of a two-population symmetric coevolutionary algorithm as:

$$x_i' = \left[\frac{(A\vec{y})_i}{\vec{x}\cdot A\vec{y}}\right] x_i \qquad (2)$$

$$y_j' = \left[\frac{\left(A^T\vec{x}\right)_j}{\vec{y}\cdot A^T\vec{x}}\right] y_j, \qquad (3)$$

where $\vec{x}'$ and $\vec{y}'$ represent the new population distributions for the next generation. This formalism assumes a variety of things about the underlying algorithm it models. First, it is assumed that an individual's fitness is assessed through pair-wise collaborations with *every* member of the cooperating population. We call this idea *complete mixing*. Second, the modeled algorithm employs fitness proportionate selection and does not consider the effects of genetic operators. Finally, the model assumes that both populations are updated in parallel from the state of their previous generation.

Dynamical systems analysis has provided a fair amount of insight regarding the fixed points in this system and their stability. For example, the basis vector associated with the maximal payoff value is a stable attracting fixed point of the system. In particular, it is a Nash equilibrium: a set of collaborating individuals that has the property that if any *one* individual is changed, the net reward for all of them will decrease. However, it is also the case that there can be other suboptimal Nash equilibria that attract trajectories [25]. Indeed, in a CCEA it is possible that most, if not all, trajectories may be pulled toward these suboptimal points. In contrast, trajectories of a simple EA with fitness proportionate selection in the ideal infinite population model will always be attracted to the basis vector associated with a unique global maximum [22].

Formally measuring the sizes of the basins of attraction of limiting behaviors of this dynamical system is difficult; however, we will estimate this information empirically using *rain gauge measures* [24] in the following way. An initial point is selected uniformly at random from the product simplexes, a trajectory through the space is computed beginning with the initial point and iterating the system model some large number of times, then the limiting behavior of the trajectory is examined. In our case, all trajectories move to the basis vectors, so we maintain a histogram corresponding to these points. If the trajectory seems to have converged "very close" to a particular basis vector[2], we increment its value in the histogram. We then repeat this process some large number of times and compute the ratios of the convergence for each basis vector.

We used this method to examine the behaviors of the two-population EGT model on the two-component MTQ problem under varying conditions affecting the relative widths of the two peaks. In this case, there were eight possible component values for each population. Obtaining trajectories from the analytical model is intractable for very large spaces, but by using a scaled-down version of the problem, we are able to compute the degree of robustness exactly for the solutions associated with the two optima. We allowed the model to iterate 2000 times for each random initial condition and performed this study for 5000 independent initial points. Table 2 shows relevant parameters for instantiating the problem, as well as relevant measures about the sub-optimal peak (f_1): the area under the peak (*Coverage*[3]), the computed degree of robustness for solutions at the sub-optimal peak (*DoR 1*), the computed degree of robustness for solutions at the optimal peak (*DoR 2*), and the ratio of random initial conditions attracted to the peak (*BOA Size*).

Table 2: Degree of robustness (DoR) measures and EGT rain gauge measures (BOA Size) for different instances of the MaxTwoQuadraticsproblem.

s_1	s_2	Coverage	DoR 1	DoR 2	BOA Size
0.5	2.0	0.2656	0.375	0.750	0.0674
1.0	1.0	0.5625	0.500	0.625	0.3526
2.0	0.5	0.7656	0.750	0.500	0.6578
4.0	0.25	0.9219	0.875	0.375	0.7164

The important thing to note about these results is that there is strong relationship between the robustness measure we defined in the previous section and the relative size of the basin of attraction of the system. Evidently, trajectories in the formal model of a traditional CCEA are attracted to solutions that obey a robustness property that is at least fairly similar to the one we've defined.

4.3 Experimentation with Real CCEA

The previous subsection used validation studies on a formal model of coevolution to gain intuition about the nature of robustness as we are measuring it and its effect on the dynamics of a coevolutionary algorithm. Here we use an experimental framework to justify the hypothesis that the run-time trajectories of real CCEAs are attracted by robust solutions, while in traditional evolutionary computation this is not necessarily so.

4.3.1 Experimental Setup

We divide our experimental landscape broadly into two categories: experimental groups involving studies of a CCEA and groups involving studies of a traditional EA. Both algorithms use a real-valued representation with Gaussian mutation and fitness proportionate selection. Mutation utilizes a separate standard deviation associated with each gene of each individual in the population, where $\sigma \in [0.05, 0.5]$ and is initialized to 0.5. As in [1], these distributions are adapted using the following procedure. Every generation a value τ is produced for each individual as follows:

$$\tau = Gauss(0, \frac{1}{\sqrt{2n}}),$$

where n is the number of genes. Then, for each individual, the σ associated with each of its genes is updated with the

[2]We consider "very close" to be a Euclidean distance of less then 10^{-4} in the $\Delta^n \times \Delta^n$ space.

[3]Here, the area can be computed exactly by enumerating the 8×8 space. In the more realistic experiments that follow, it is estimated using a discrete sampling grid.

equation:

$$\sigma_i' = \sigma_i e^{\tau + Gauss(0, \frac{1}{\sqrt{2\sqrt{n}}})}.$$

The EA trials used a population size of 100 and were run for 200 generations, while the CCEA trials used two populations each of size 100, but were run for only 100 generations for consistency in the total number of evaluations. The CCEA represents each variable of the optimization problem separately, one per population, and chooses a single random collaborator from the alternate population each evaluation. The populations were evolved in sequential iteration.

For each of these algorithms, twelve experimental groups were created, one for each of twelve different instances of the MTQ problem produced by ranging the variables $s_1 \in \{1.0, 2.0, 4.0\}$ and $s_2 \in \{0.25, 0.5, 1.0, 2.0\}$. The other parameters of MTQ remained consistent with the values stated above. There were 50 trial runs for all groups.

4.3.2 Examining Convergence Results

To get a sense for how attractors in the problem affect convergence results of real algorithms, we need two pieces of information: What is the ratio of runs that converge at or near the suboptimal peak, and what is the robustness of that peak. In the former case, we determined to which peak the best-of-final-generation solutions were closest by simple Euclidean distance in the domain space. In the latter case, we elicited a sample 128×128 payoff matrix and computed an approximation of the robustness at each peak[4], paying particular attention to the local peak. The following table shows these robustness results for each of the problem instances. Each cell contains a pair, the first value in the pair refers to the approximate robustness value of potential solutions at the suboptimal peak, the second refers to the optimal peak. The values are rounded to two decimal places.

Table 3: Robustness approximations for two potential solutions to different MTQ problem instances (peak 1, peak 2).

		s_2			
		0.25	0.5	1.0	2.0
	1.0	(0.66, 0.38)	(0.61, 0.48)	(0.54, 0.54)	(0.46, 0.61)
s_1	2.0	(0.85, 0.33)	(0.67, 0.45)	(0.56, 0.52)	(0.48, 0.59)
	4.0	(0.88, 0.28)	(0.75, 0.40)	(0.57, 0.49)	(0.48, 0.58)

Here some care is needed since there are multiple effects of changing the aforementioned problem parameters. One important effect is that the coverage of the two peaks in the domain space is greatly altered. In the extreme cases, it produces problems in which the suboptimal peak dominates a large percentage of the domain of the landscape and can affect the algorithms because of simple initialization issues. It is likely both algorithms will be affected by such issues, and it is also the case the coverage and robustness are reasonably well correlated. To protect against this complicating our analysis, we include coverage as a variable in our statistical comparisons. We approximate the coverage of the suboptimal peak for each problem instance using the same 128×128 matrix used to approximate robustness at the two peaks. The following table details this information. Again, the values are rounded.

[4]The real algorithms operate in a real-valued space (i.e., much larger than 128×128). The reduced matrix approximates the robustness values of the two potential solutions.

Table 4: Domain coverage of peak 1 for different MTQ problem instances.

		s_2			
		0.25	0.5	1.0	2.0
	1.0	63%	60%	55%	45%
s_1	2.0	86%	80%	72%	60%
	4.0	93%	86%	76%	66%

The convergence results for the two algorithms are shown in Figure 1. These scatter plots show the degree of robustness of peak 1 on the x-axis (from Table 4) versus ratio of runs converging to peak 1 on the y-axis for each of the 12 experimental groups for both algorithms. The CCEA results are shown as black dots, the EA as grey diamonds.

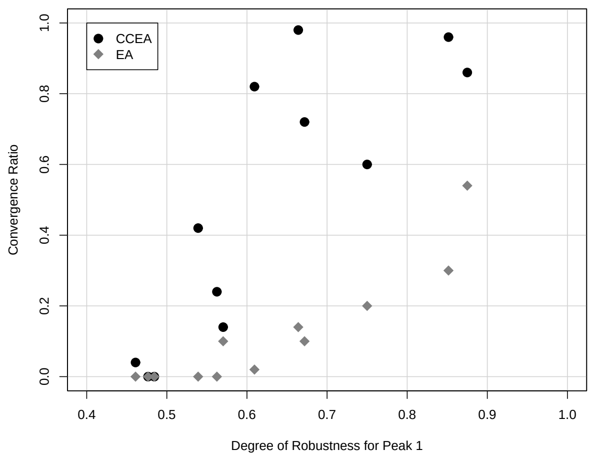

Figure 1: Scatter plots showing degree of robustness versus convergence ratios for each of the 12 experimental groups for both algorithms.

Our hypothesis is that the robustness properties of potential solutions in the problem will affect the result of the CCEA, while they will have little or no effect on the EA. To answer this question, we use a simple multivariate regression method considering both variables (robustness and coverage) as indicators of the ratio of convergence to peak 1, the suboptimal peak. With 95% certainty, we can reject the hypothesis that the robustness measure contributes to predicting the convergence results in the case of the EA; however, we cannot reject that both coverage and the robustness measure explain the convergence results in the case of the CCEA. In other words, in the combined prediction model, the robustness measure's contribution is insignificant to predicting the convergence results in the EA groups, while in the case of the CCEA, coverage alone is insufficient to explain the convergence results.

4.3.3 Examining Run-Time Behaviors

In addition to this type of correlation study, it is useful to consider the actual run-time behavior of the algorithms. We do this by examining the robustness of the *candidate* best solution in the population(s) at each generation for each algorithm on two particular problem instances. Since we will now need to be able to estimate the robustness of an arbitrary candidate solution, and since the practical algorithms search a vastly larger space than the 128×128

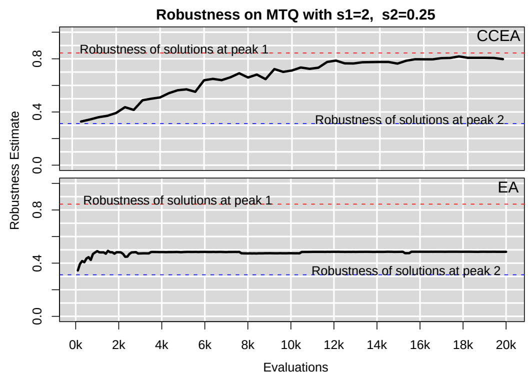

Figure 2: Run-time robustness results for the CCEA (top panel) and EA (bottom panel) on the MTQ problem with $s_1 = 2.0,\ s_2 = 0.25$. Curves represent average estimated robustness of the x component over 50 runs. The dashed lines show the robustness of the potential solution at each peak.

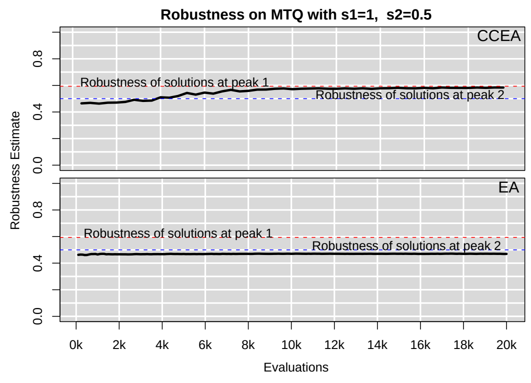

Figure 3: Run-time robustness results for the CCEA (top panel) and EA (bottom panel) on the MTQ problem with $s_1 = 1.0,\ s_2 = 0.5$. Curves represent average estimated robustness of the x component over 50 runs. The dashed lines show the robustness of the potential solution at each peak.

matrix we used to compute the approximate robustness of known potential solutions at the two peaks, we now need to *estimate* the robustness of candidate solutions considered by the search algorithms during the run. We do this by selecting the best individual from each generation and sampling the space using a discrete 100×100 grid. From the perspective of the x component, we examine its robustness using this grid in the following way. We pair the value of x with the first sampled y value and compute its fitness. We then compute the average fitness of 100, evenly spaced values of x with that y sample. If the original pairing results in a greater fitness than the average, it counts toward the robustness of the x component value. We do this for all 100 sampled values of y and compute the ratio of the samples that meet the criterion. We can also perform the complementary and symmetric process for computing the the robustness of y component values.

Above we see the results of runs on the $s_1 = 2.0,\ s_2 = 0.25$ and the $s_1 = 1.0,\ s_2 = 0.5$ MTQ problem for each of the two algorithms. We selected these two examples because, while they do not represent extreme values of the space of our MTQ problem instances, they do illustrate two different situations: One in which the robustness of the two peaks is very different and one in which they are much closer, though, in both cases the robustness of the solution at peak 1 is higher. The curves in the graph are produced by calculating the estimated robustness of the best current solution from the perspective of the x component at each generation and averaging this value across 50 independent trials. We provide points of reference by plotting the values reported in Table 4, the robustness approximations of the potential solutions at each of the two peaks, as labeled horizontal dashed lines in the graphs. We also examined the y component, and the results (not shown) are consistent with what is shown.

From these graphs, it is clear that the CCEA *is* exploiting the robustness property during its search. The degree to which the EA is doing so is unclear. While there is some minimal improvement during the early stages in Figure 2, this improvement may (again) have more to do with ancillary initialization effects due to other properties affected by the change in the s_1 and s_2 parameters. In any event, in all of the cases we examined, there is either no progress toward a greater robustness value in the EA, or there is a minimal progress made initially that levels out quickly after initialization. Save for extreme cases, in all experiments we examined, the CCEA made constant progress against the robustness measure.

5. CONCLUSIONS AND FUTURE WORK

While CCEAs may not be suitable for many static function optimization problems, in our experience they have been very useful for the kinds of multiagent learning problems we face in our lab. The algorithms we use seem appropriate for finding behaviors for team members that result in good, though not necessarily optimal performance. More importantly, the performance of member behaviors often exhibit a kind of robustness to changes in other team members. Given that this observation is consistent with, but not explained by, the prevailing theory for CCEAs, we decided to take a closer look at why this might be so.

To begin, we took inspiration from existing literature in the operations research community to help clarify what might be meant by "robustness". We provided a general framework, based on two key elements that allows researchers to instantiate a specific and clear definition of robustness for their situation. First, the solution space for the problem is partitioned into two subspaces. Second, a criterion is provided with which to compare variations of potential solutions. With these pieces, one can discuss the degree of robustness of a particular value in one subspace over changes in the other subspace with respect to the specified criterion. We provided a particular criterion asserting that a given component value in the first partition is robust if it maximizes the number of pairings with component values from the second partition that have a better than average performance across alternative strategies from the first.

Next, we looked to a known evolutionary game theoretic dynamical systems model of an idealized CCEA for intuition. We found that, unlike traditional fitness-proportionate based idealized EAs, trajectories in such a system appear to be *attracted* to population states that concentrate around solutions with the robustness property we defined, whether or not these solutions correspond with optimal objective values. Given a choice between a the global optimum and an alternative optimum that has a lower objective value but a sufficiently high degree of robustness, the idealized CCEA will more often be drawn by the robust solution.

We implemented an EA and a CCEA and applied them to a common class of problems from literature. We found that, while things are as always more complicated in more realistic settings, even in praxis the robustness property of a potential solution seems to be a definite indicator of where CCEAs will converge, while not necessarily being so for the EA we studied. Moreover, when looking at run-time estimates of robustness we noted that the CCEA progresses toward the optima with a high, though not optimal, objective value but with a higher degree of robustness, while the EA tends to favor objective performance with little to no consideration for the property.

While we recognize that these results are only the first steps, we believe it is constructive to view many CCEAs as optimizers of robustness. Understanding what this entails will help us apply the algorithms more effectively for problems where robustness is an important property of the desired solution. We are currently working on a formal mathematical proof that the EGT model of the CCEA is, in fact, drawn to points that maximize degree of robustness. We will also continue our research by experimenting with more realistic multiagent problems, offering a more foundational understanding for how cooperative coevolution solves such problems.

Acknowledgments

This work was performed under Office of Naval Research Work Orders N0001406WX20006 and N0001406WX30002.

6. REFERENCES

[1] T. Bäck and H.-P. Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[2] A. Bucci and J. Pollack. On identifying global optima in cooperative coevolution. In *Proceedings of the 2005 Genetic and Evolutionary Computation Conference*, pages 539–544. ACM Press, 2005.

[3] D. Cliff and G. F. Miller. Tracking the red queen: Measurements of adaptive progress in co–evolutionary simulations. In *Proceedings of the Third European Conference on Artificial Life*, pages 200–218. Springer–Verlag, 1995.

[4] L. Dias and J. Clmaco. On computing electres credibility indices under partial information. *Journal of Multi-Criteria Decision Analysis*, 8:74–92, 1999.

[5] S. G. Ficici. *Solution Concepts in Coevolutionary Algorithms*. PhD thesis, Brandeis University, Boston, MA, 2004.

[6] J. Hofbauer and K. Sigmund. *Evolutionary Games and Population Dynamics*. Cambridge University Press, 1998.

[7] P. Husbands and F. Mill. Simulated coevolution as the mechanism for emergent planning and scheduling. In *Proceedings of the Fourch International Conference on Genetic Algorithms*, pages 264–270. Morgan Kaufmann, 1991.

[8] E. Jen. Santa Fe Institute program on Robustness. http://discuss.santafe.edu/robustness/, 2002.

[9] E. Jen. Stable or robust? what's the difference? *Complexity*, 8(3):12–30, 2003.

[10] P. Kouvelis and G. Yu. *Robust discrete optimization and its applications*. Kluwer, Dordrecht, 1997.

[11] J. Maynard-Smith. *Evolution and the Theory of Games*. Cambridge University Press, 1982.

[12] D. Moriarty and R. Miikkulainen. Forming neural networks through efficient and adaptive coevolution. *Evolutionary Computation*, 5(4):373–399, 1997.

[13] J. Mulvey, R. Vanderbei, and S. Zenios. Robust optimization of large-scale systems. *Operations Research*, 43:264–281, 1995.

[14] L. Panait, S. Luke, and R. P. Wiegand. Biasing coevolutionary search for optimal multiagent behaviors. *IEEE Transactions on Evolutionary Computation*, (To appear), 2006.

[15] M. Potter. *The Design and Analysis of a Computational Model of Cooperative CoEvolution*. PhD thesis, George Mason University, Fairfax, Virginia, 1997.

[16] M. Potter and K. De Jong. The coevolution of antibodies for concept learning. In *Proceedings of the Fifth International Conference on Parallel Processing in Nature*, pages 530–539. Springer-Verlag, 1998.

[17] M. Potter and K. De Jong. Cooperative coevolution: An architecture for evolving coadapted subcomponents. *Evolutionary Computation*, 8(1):1–29, 2000.

[18] M. Potter, A. Schultz, and L. Meeden. Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence*, pages 1337–1343. Morgan Kaufmann, 2001.

[19] M. A. Potter, K. A. De Jong, and J. J. Grefenstette. A coevolutionary approach to learning sequential decision rules. In *Proceedings from the Sixth International Conference on Genetic Algorithms*, pages 366–372. Morgan Kaufmann, 1995.

[20] B. Roy. A missing link in or-da: robustness analysis. *Foundations of Computing and Decision Sciences*, 23:141–160, 1998.

[21] P. Vincke. Robust solutions and methods in decision-aid. *Journal of Multi-Criteria Decision Analysis*, 8:181–187, 1999.

[22] M. Vose. *The Simple Genetic Algorithm*. MIT Press, 1999.

[23] R. P. Wiegand. *An Analysis of Cooperative Coevolutionary Algorithms*. PhD thesis, George Mason University, Fairfax, Virginia, 2003.

[24] R. P. Wiegand, W. Liles, and K. De Jong. Analyzing cooperative coevolution with evolutionary game theory. In *Proceedings of the 2002 Congress on Evolutionary Computation (CEC)*, pages 1600–1605. IEEE Press, 2002.

[25] R. P. Wiegand, W. Liles, and K. De Jong. Modeling variation in cooperative coevolution using evolutionary game theory. In *Foundations of Genetic Algorithms VII*, pages 231–248. Morgan Kaufmann, 2002.