An Empirical Analysis of Collaboration Methods in Cooperative Coevolutionary Algorithms

R. Paul Wiegand
George Mason University
Computer Science Department
Fairfax, VA 22030
paul@tesseract.org

William C. Liles*
Central Intelligence Agency
Washington, DC 20505
wliles@gmu.edu

Kenneth A. De Jong
George Mason University
Computer Science Department
Fairfax, VA 22030
kdejong@gmu.edu

Abstract

Although a variety of coevolutionary methods have been explored over the years, it has only been recently that a general architecture for cooperative coevolution has been proposed. Since that time, the flexibility and success of this cooperative coevolutionary architecture (CCA) has been shown in an array of different kinds of problems. However, many questions about the dynamics of this model, as well as the efficacy of various CCA-specific choices remain unanswered. One such choice surrounds the issue of how the algorithm selects collaborators for evaluation. This paper offers an empirical analysis of various types of collaboration mechanisms and presents some basic advice about how to choose a mechanism which is appropriate for a particular problem.

1 Introduction

In recent years there has been a growing interest in coevolutionary algorithms as interesting and useful extensions to the more traditional Evolutionary Algorithms (EAs). The important difference in moving to coevolutionary algorithms is that the fitness of an individual is a function of the other individuals in the population. Two basic classes of coevolutionary algorithms have been developed: competitive coevolution in which the fitness of an individual is determined by a series of competitions with other individuals (see, for example, Rosin and Belew (1996)), and cooperative coevolution in which the fitness of an individual is determined by a series of collaborations with other individuals (see, for example, Potter and De Jong (2000)). Both types of coevolution have been shown to be useful for solving a variety of problems.

In this paper our focus is on cooperative coevolutionary algorithms (CCAs). A standard approach to applying CCAs to a problem is to identify a natural decomposition of the problem into subcomponents. Each component is assigned to a subpopulation, such that individuals in a given subpopulation represent potential components to the greater problem. Then each component is evolved simultaneously, but in isolation to one another. In order to evaluate the fitness of an individual from a given subpopulation, collaborators are selected from the other subpopulations in order to form a complete solution.

While the CCA has shown definite promise on various problems, there is still a lot that we do not know about how the model works. One major question is still the issue of how collaborators are chosen. It is clear from early work that problem characteristics are connected with this choice. For instance, problem landscapes with strong inter-activity between components seem to require less greedy methods for selection of collaborators.

In this paper we examine this choice by looking at three main aspects of collaboration: collaborator selection pressure, the number of collaborators for a given evaluation, and credit assignment of fitness in evaluation when using multiple collaborators. These attributes are examined by a series of experimental studies on a variety of different function optimization problems.

In the next section we will discuss some background about coevolutionary approaches. The third section will describe the architecture in more detail, as well as illustrating the various choices surrounding collaboration. Then we will describe the experiments that were run and the results obtained. Finally, we will discuss our conclusions and try to offer some practical advice about how to do collaboration in the CCA in light of particular problems.

*This material has been reviewed by the CIA. That review neither constitutes CIA authentication or information nor implies CIA endorsement of the author's views.

†From Proceedings of the Genetic and Evolutionary Computation Conference, 2001. ©Morgan Kaufmann Publishers

2 Background

Although evolutionary algorithms (EAs) have been with us for half a century, significant research into the use of coevolutionary systems did not really begin until the early 1990's. From the start early research focused on applications of complex tasks, such as competitive approaches using a parasite-host relationship as a model to coevolve sorting networks and problem sets (Hillis, 1991), and cooperative approaches for coevolving job-shop schedules (Husbands and Mill, 1991).

However, although both cooperative and competitive approaches were explored from the beginning, most research since that time has dealt with applications of competitive approaches. Most popularly competitive coevolution has been applied to game playing strategies (Rosin and Belew, 1995, 1996; Rosin, 1997; Pollack and Blair, 1998). Additionally Angeline and Pollack (1993) demonstrate the effectiveness of competition for evolving better solutions by developing a concept of competitive fitness to provide a more robust training environment than independent fitness functions. Competition was also successfully harnessed by Schlierkamp-Voosen and Mühlenbein (1994) to facilitate strategy adaptation in breeder genetic algorithms.

More recently a variety of coevolutionary methods have been applied to machine learning problems. There has been particular attention to neural network coevolution (Paredis, 1994; Juillé and Pollak, 1996; Mayer, 1999; Potter and De Jong, 2000). Additionally, some work in using coevolutionary algorithms for concept learning has been done (Potter and De Jong, 1999).

Moreover, there have been recent attempts to lay out general frameworks for coevolutionary models (Potter and De Jong, 1994; Paredis, 1996). However, attempts to understand the dynamics of these frameworks have been few and far between. Some basic empirical work is laid out by Potter and De Jong (1994) and Potter (1997), indicating that there is a possible link between variable inter-activity and collaboration selection. Even more recently, some basic theoretical work has been done to take ideas from simple genetic algorithm theory provided by Vose (1999), and apply it to coevolution (Ficici and Pollack, 2000). This work explores the mechanics of a simple competitive coevolutionary algorithm from a game theoretic viewpoint.

These questions of coevolutionary dynamics are not academic. The question of selecting collaborators for evaluation, for instance, has not only been an issue for our application activities, but has also cropped up with other researchers who are applying the techniques to problems such as inventory control optimization Eriksson and Olsson (1997). Indeed, since this is a key element of the success of applying this cooperative coevolution architecture (CCA), we believe it merits particular attention in order to improve our ability to apply CCAs in the future.

3 Coevolution and Collaboration

When applying a CCA to a particular problem, a standard approach is to decompose the problem into subcomponents and assign each subcomponent to a subpopulation. These subpopulations may or may not be homogeneous with respect to the representation used or the EA being used to evolve a particular subcomponent.

Evolution proceeds independently, except for evaluation. Since any given individual from a subpopulation represents only a subcomponent of the problem, *collaborators* will need to be selected from the other subpopulations in order to assess fitness. Each generation, all individuals belonging to a particular subpopulation have their fitness evaluated by selecting some set of collaborators from other subpopulations to form complete solutions. Afterward, the CCA proceeds to the next subpopulation, which will in turn draw collaborators from each of the other subpopulations. A simple algorithm of this process is outlined below in figure 1.

$gen = 0$
for each species s **do**
  $Pop_s(gen) =$ initialized population
  evaluate($Pop_s(gen)$)
while not terminated **do**
  $gen++$
  for each species s **do**
    $Pop_s(gen) \leftarrow$ select($Pop_s(gen-1)$)
    recombine($Pop_s(gen)$)
    evaluate($Pop_s(gen)$)
    survive($Pop_s(gen)$)

Figure 1: The structure of a Cooperative Coevolutionary Algorithm (CCA).

Computing the fitness of individuals in a coevolutionary system can be done in a variety of ways. Some competitive coevolutionary algorithms perform bipartite evaluations, applying each individual in one population to each in the other (Hillis, 1991). Additionally, it is not uncommon for single population competitive fitness models to perform exhaustive pair-wise evaluations (Axelrod, 1989). Such approaches can be computationally expensive in multi–population models, since the number of objective function evaluations used for each assessment of fitness grows exponentially by the number of species. Less expensive approaches, such as single elimination tournaments have also been addressed (Angeline and Pollack, 1993).

Alternatively, early work in the CCA model suggests two methods for selecting collaborators for the purpose of fitness evaluation:

CCA-1 Choose the best individuals from alternative subpopulations, as defined by fitness obtained from the last evaluation process of that group.

CCA-2 Select two individuals: the best and a random individual. Evaluate both with the current individual and use the higher objective function value for the current individual's fitness score.

We could of course imagine many more such methods. Rather than speculate about potential collaboration choices, however, it is more useful to define some basic attributes of this choice. In our opinion, there are three such attributes:

- The degree of greediness of choosing a collaborator (*collaborator selection pressure*).
- The number of collaborators per subpopulation to use for a given fitness evaluation (*collaboration pool size*).
- The method of assigning a fitness value given multiple collaboration-driven objective function results (*collaboration credit assignment*).

We will attempt to address all three of these attributes in our research.

3.1 A Clearer Picture of Collaboration

Since an individual represents only a *subcomponent* of a problem solution, collaborator subcomponents from each of the other subpopulations must be assembled to form a complete solution. We call the process of choosing these collaborator subcomponents *collaborator selection*. To do this, we can use the last evaluated fitness scores of the individuals in the alternative subpopulations to bias how we choose these collaborators. The degree of bias in this choice is what we are calling *collaborator selection pressure*.

This newly assembled complete candidate solution (a *collaboration*) can now be plugged into the objective function and assigned a collaboration score. If only one collaborator from each subpopulation is selected, there will only be a single collaboration score, and this score may be used as the fitness value.

However, we may choose to try different combinations of collaborators from the other subpopulations. In which case evaluation of an individual will consist of multiple collaborations. The number of collaborators selected from each subpopulation is what we call the *collaboration pool size*. Since each of these collaborations will have their own collaboration score from the objective function, these multiple scores must be in some way resolved to a single fitness value (*collaboration credit assignment*).

4 Experiment Methods

4.1 Fitness Landscapes

For our initial experimental studies on collaboration methods we elected to study simple function optimization problems. These types of problems are nice for the CCA, since a natural decomposition of the problem is very straightforward. Each subpopulation represents a particular variable of the function. In all cases, we chose to examine only two variable landscapes, since increasing the dimensionality creates a combinatorial problem and raises questions about how multiple collaborators are applied. Future work on this matter is needed.

We use three fitness functions: Rosenbrock, Rastrigin, and a quadratic which is not directly aligned with the axes. The first two were chosen to be consistent with the Potter and De Jong (1994) study. They represent two difficult problems, one of which is not linearly separable (Rosenbrock) and the other which is linearly separable (Rastrigin). The third function was chosen because, although it is a simple problem conceptually, the lack of axis alignment has been shown to introduce problems for certain kinds of evolutionary algorithms (Salomon, 1996). A table summarizing the functions used, as well as the constraints of those functions is show in table 1. In all cases the functions were to be minimized.

4.2 EA Characteristics

As previously stated, subpopulations of the CCA are homogeneous in our study. Again, where possible EA characteristics remain similar to previous studies. In all cases, the details of the evolutionary mechanisms are as follows:

representation:	binary (16 bits per function variable)
selection:	fitness proportionate with linear scaling
genetic operators:	two-point crossover & bit-flip mutation
mutation probability:	1 / chromosome length
crossover probability:	0.6
(sub)population size:	100
termination criteria:	100,000 function evaluations

Table 1: Function Test Suite

Function	Bounds	Name
$f_1(x_1, x_2) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	$-2.048 \leq x_i \leq 2.048$	Rosenbrock
$f_2(x_1, x_2) = 6 + \sum_{i=0}^{2} x_i^2 - 3.0cos(2\pi x_i)$	$-5.12 \leq x_i \leq 5.12$	Rastrigin
$f_3(x_1, x_2) = x_1^2 + (x_1 + x_2)^2$	$-65.536 \leq x_i \leq 65.536$	Off-axis quadratic

5 Experimental Results

We used the experimental setup described above to run a large number of experiments in order to see if there were clear patterns indicating how best to design a CCA collaboration method. Recall that designing a collaboration method involves three decisions: how to assign fitness when there are multiple collaborations, how to choose collaborators, and how many collaborators to choose. Each of these issues is explored in the following subsections.

5.1 Collaboration Credit Assignment

We explored three methods for assigning an eventual fitness score to individuals who have had multiple collaborative function evaluations. These methods are as follows:

Optimistic: The more traditional method of assigning an individual a fitness score equal to the value of its best collaboration.

Hedge: Assign an individual a fitness score equal to the *average* value of its collaborations as is generally done in competitive coevolution.

Pessimistic: Assigning an individual a fitness score equal to the value of its worst collaboration.

The intuition for the latter option is that perhaps it might be best to use a "safe" credit assignment that rewards an individual only as well as its weakest collaboration. However, in all of our experiments this never turned out to be the case. In fact both the pessimistic and hedge strategies consistently resulted in significantly poorer performance in our studies, generally by several orders of magnitude. Some typical examples of these results can be seen in Table 2, involving minimizing a two variable Rosenbrock function when selecting two and three collaborators.

As a consequence of this consistent pattern, the remainder of the experiments discussed in this paper will only use the optimistic approach for credit assignment.

5.2 Collaboration Selection Pressure

The next attribute for deciding collaboration mechanics is the degree of greediness of choosing a collaborator.

Table 2: Example collaboration credit assignment results for the Rosenbrock minimization problem. The number represent averages across 50 trials. Collaborators are chosen at random.

Collaborator Poolsize	Credit Assignment	Result
2	Optimistic	43.35
2	Hedge	$10,413.6$
2	Pessimistic	$7,063.22$
3	Optimistic	25.82
3	Hedge	$76,870$
3	Pessimistic	$46,843.5$

CCA-1 uses a very greedy method, selecting the best individual from the previous generation. CCA-2, however, weakens this pressure somewhat by using a two collaborator mechanism in which the second collaborator is chosen at random. It is still quite greedy however, since the best individual is still used. We can imagine weakening this pressure even more by allowing for different combinations of selection mechanisms aside from selecting the best. Tables 3, 4, and 5 show results for all three functions using various combinations of collaborator pool size two, as well as three selection mechanisms: best, random, and worst.

Notice that no improvements are obtained on the Rosenbrock function *regardless* of whether the best individual is included as one of two collaborators. This finding is consistent with ANOVA at 95% confidence. In the Rastrigin function and the quadratic functions however, there is a clear significant advantage to using the best individual as one of the collaborators. We will return to this later in the paper.

We can weaken selection even further, as well as provide ourselves with a way of controlling the degree of collaborator selection pressure by using previous generation evaluation results to bias a non-deterministic choice of current collaborations. We use a method similar to tournament selection, varying tournament sizes from 1 (random selection) to 4, which is a fairly strong bias. More extreme sizes were used, but are not presented in detail in this paper.

Table 3: Rosenbrock minimization results using various combinations of selection choices for each of two collaborators: best, worst, and random. These show averages of 50 trials, as well as 95% confidence intervals.

Rosenbrock	**Random**	**Best**
Worst	56.51 ± 44.24	57.67 ± 48.49
Random	37.82 ± 24.29	68.28 ± 88.43

Table 4: Rastrigin minimization results using various combinations of selection choices for each of two collaborators: best, worst, and random. These show averages of 50 trials, as well as 95% confidence intervals.

Rastrigin	**Random**	**Best**
Worst	6.43 ± 1.93	0.54 ± 0.03
Random	3.46 ± 1.06	0.54 ± 0.05

Table 5: Off-Axis quadratic minimization results using various combinations of selection choices for each of two collaborators: best, worst, and random. These show averages of 50 trials, as well as 95% confidence intervals.

Quadratic	**Random**	**Best**
Worst	295.07 ± 199.19	2.85 ± 5.45
Random	136.63 ± 82.96	1.21 ± 0.93

This method should not be confused with that of selecting multiple collaborators. In the case where the collaboration pool size is only one, we can still have large tournament sizes without affecting the fact that only a single objective function application is required to evaluate an individual. This tournament-style collaborator selection method is used merely to leverage previous generation fitness values as a way to bias our search for a collaborator for the current evaluation. Of course the method is still reasonable for larger collaboration pool sizes, and again the size of the tournament will not impact the number of collaborations that are ultimately formed.

Figure 2 shows the results for the minimization experiments of the Rosenbrock function. The x-axis represents fitness scores. The points on the plot are averages of 50 trials for each experimental group. The whiskers show the 95% confidence intervals of these groups. Results for the basic GA, CCA-1, and CCA-2 groups are shown at the top for baseline comparison. The remaining four panels show the results for groups which were run using increasing collaboration selection pressure.

Notice that varying the pressure with this tournament-like collaborator selection method makes very little difference in the overall performance of the CCA. The f_1, f_2, and f_3 graphs show a similar story in this respect. Although the graphs in this paper show results for conservative ranges of this parameter, a range of extreme values (such as tournament sizes limiting toward the population sizes, and tournament selection of collaborators with the worst fitness score, etc.) were also run. These trends bear out even at extreme values.

More interesting perhaps is the fact that with the non-linear case, collaborator selection method neither helps, nor creates a performance degradation unless the selection bias is so strong that it effectively reduces the number of collaborators used (for extreme tournament sizes). Indeed, recall from Table 3 that using one best and one random (or worst) individual as collaborators in a two-way collaboration resulted in no significant performance difference.

Clearly it is not entirely the case that collaboration selection pressure is unimportant, however, since CCA-1 and CCA-2 do quite well against the linearly separable Rastrigin and the off-axis quadratic function compared to those CCA algorithms employing random selection.

One hypothesis is that the CCA-1 and CCA-2 algorithms' use of the most extreme collaboration selection pressure creates a search which is similar in behavior to a line-search. Therefore Rosenbrock non-linearities create a problem for which this bias gives us little or no advantage. In the case of the other two functions, this simple line-search type of behavior provides a strong bias which is well suited to solving these problems.

5.3 Collaboration Pool Size

The most dramatic effect on the success of the CCA was clearly the number of collaborators one uses. To some extent, computationally speaking this is an unfortunate, though not surprising, result since increasing the number of collaborators can significantly increase overall computation time—a problem which is combinatorial with the number of subpopulations.

Again look at figures 2, 3, and 4. In almost all cases increasing the number of collaborators used assisted the performance of the algorithm. In fact, although the relaxation of the greedy collaboration method hinders the CCA with respect to the GA on the Rastrigin function, increasing the collaborative pool size to 5 with random collaborator selec-

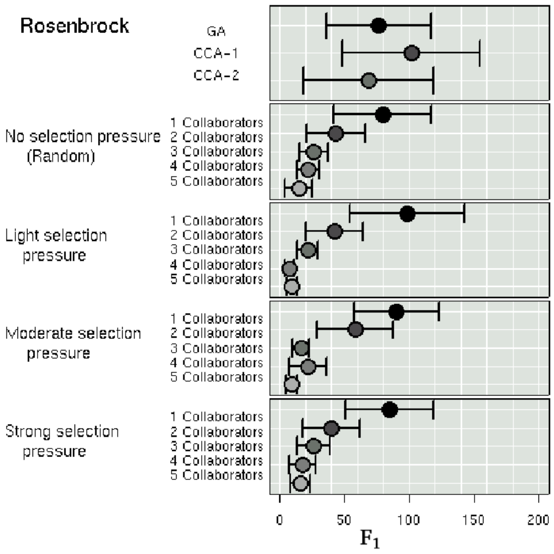

Figure 2: Results for Rosenbrock (f_1) minimization experiments. The x-axis represents the final reported result from the EA after 100,000 function evaluations. The points plotted are averages of 50 trials, and the whiskers show the 95% confidence intervals.

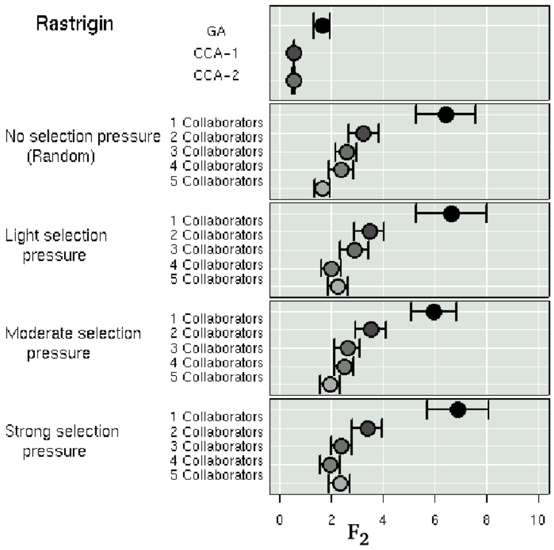

Figure 3: Results for Rastrigin (f_2) minimization experiments. The x-axis represents the final reported result form the EA after 100,000 function evaluations. The points plotted are averages of 50 trials, and the whiskers show the 95% confidence intervals.

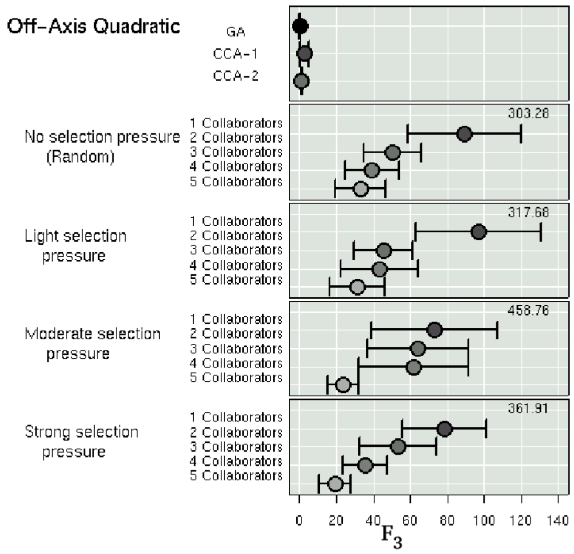

Figure 4: Results for the off-axis quadratic (f_3) minimization experiments. The x-axis represents the final reported result from the EA after 100,000 function evaluations. The points plotted are averages of 50 trials, and the whiskers show the 95% confidence intervals.

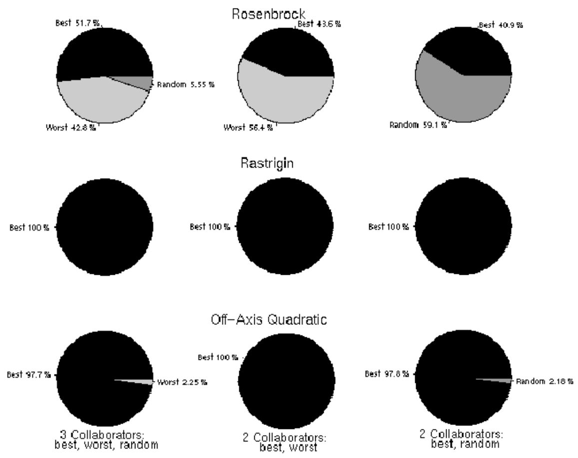

Figure 5: The contributions of collaborators were tracked using three different collaboration mechanisms for each function (best, worst, and random; best and worst; and best and random). The number of times each collaborator is responsible for yielding the better fitness score is illustrated as ratios in the above chart. Averages across 50 trials were used.

tion returns CCA performance to at least an insignificant difference from that of the GA.

While our research does not address the issue of how large the collaboration pool size should be for a given problem, it does suggest that a relatively conservative adjustment from one to two collaborators will frequently yield substantial benefit. Indeed, in all three functions this change is statistically significant for the data shown in our figures.

Of course even such a "conservative" adjustment from one to two collaborators is disappointing news in terms of computational complexity. Whether some sampling technique can be used to reduce the combinatorial problem remains to be seen.

6 Further Analysis

In order to look more closely at what kind of use the CCA is making of its collaborators, we decided to setup some multi-collaborator runs and track the frequency with which the CCA makes use of any particular collaborator for fitness assessment. We did this by setting up three groups for each function, one requiring three collaborators, and two requiring just two. In the first group the best, worst, and random collaborators were selected and the function was evaluated with each. The second group selects the best and worst collaborator and the third selects best and random. Since we are using an optimistic strategy for credit assignment, we kept track of the number of times each collaborator produces the best resulting fitness score. Figure 5 on page 6 shows these ratios for these groups for all three functions.

The Rastrigin function always uses the best collaborator. Given the premise that using the best individual for collaboration gives us something like a simple line-search, it is not surprising that the best collaborator is always the one used by the algorithm during evolution. Even the off-axis quadratic makes predominant use of the best individual, although it seems evident that its own alignment properties create a need for small use of other collaborators.

What is more interesting is that not only does the Rosenbrock make use of all three, but it doesn't seem to matter whether we use a random collaborator or the worst individual as a collaborator. Curiously, use of the worst individual seems to overshadow use of the random individual in the three collaborator case. We believe this is an artifact of the properties of this particular landscape, though clearly more investigation is needed to fully explain this.

7 Conclusions and Future Work

There are several clear lessons to take home from this research. First and foremost it is evident that using an optimistic approach is generally the best mechanism for collaboration credit assignment. This may not always be true for every type of problem, but it seems that it is a very safe first guess for static objective functions.

The next question a practitioner should ask is how much non-linear interaction there is likely to be among the subcomponents with respect to fitness. If this can be reasonably assessed, it is the key to making the next decisions about collaboration. Clearly if the problem is a simple problem that is linearly separable, a greedy approach to collaborator selection is warranted. Additionally, it may well be that the number of collaborators may be limited (perhaps even to just 1).

For more complicated problems with large degrees of variable interactivity, the selection pressure of collaboration is far less important than the number of collaborators. Moreover, if computationally feasible, increasing the number of collaborators seems to benefit CCA performance in general.

However, it is also clear that there is no magic bullet, of course. The simple off-axis quadratic still perplexes the non-greedy CCA. Combining random (or worst, or arbitrary) collaborators with best collaborators against these problems resulted in no significant degradation of solution quality over the greedy CCA-1. So combining these methods (as in CCA-2 by Potter and De Jong (1994)) may be a good first stab at solving a problem when the degree of variable interactivity is unknown.

The fact that in the non-linear case collaborator selection pressure seems to be unimportant may be a clue for some resolution to the multi–collaborator combinatorial problem. It suggests that *how* you sample the collaboration space is not very important. This encourages the possibility that some simple sampling methods can give us some relief to this problem. As state earlier, more work here is needed.

Although we feel this is a good first step toward understanding how collaboration works in the CCA, much work remains. Ideally it would be nice to have some theory that allows us to find the minimal number of collaborators necessary to solve a given problem, for instance.

Even without such a theory though many questions are raised by this research which deserve attention. First of all, if the CCA makes even use of different collaborators in non-linear problems like Rosenbrock as it seems to, what kind of run-time behavior does this usage have? Is there some sort of periodicity, as one collaborator selection method dominates the other for some time, then trends reverse? Or is there some punctuation of equilibrium that occurs to cause the method to re-balance? Or perhaps the usage is random and unpredictable. Exploring this question is one of our foremost questions.

Additionally, while increasing from two to three collaborators in the Rosenbrock problem is clearly superior, CCA runs do not seem to make even distributed use of these collaborators. Our observations show that two selection methods can overshadow the third, even though the addition of a third method improves performance. We would like to answer this question in the future, as well.

References

P. Angeline and J. Pollack. Competitive environments evolve better solutions for complex tasks. In S. Forrest, editor, *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 264–270, San Mateo, CA, 1993. Morgan Kaufmann.

R. Axelrod. Evolution of strategies in the iterated prisoner's dilemma. In L. Davis, editor, *Genetic Algorithms and Simulated Annealing*. Morgan Kaufman, 1989.

R. Eriksson and B. Olsson. Cooperative coevolution in inventory control optimisation. In G. Smith, N. Steele, and R. Albrecht, editors, *Proceedings of the Third International Conference on Artificial Neural Networks and Genetic Algorithms*, University of East Anglia, Norwich, UK, 1997. Springer.

S. Ficici and J. Pollack. A game-theoretic approach to the simple coevolutionary algorithm. In *Proceedings from the Sixth Parallel Problem Solving from Nature*, pages 467–476. Springer-Verlag, 2000.

D. Hillis. Co-evolving parasites improve simulated evolution as an optimization procedure. *Artificial Life II, SFI Studies in the Sciences of Complexity*, 10:313–324, 1991.

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

H. Juillé and J. Pollak. Co-evolving interwined spirals. In L. Fogel, P. Angeline, and T. Bäck, editors, *Proceedings of the Fifth Annual Conference on Evolutionary Programming*, pages 461–468. MIT Press, 1996.

H. Mayer. Symbiotic coevolution of artificial neural networks and training data sets. In *Proceedings from the Fifth Parallel Problem Solving from Nature*, pages 511–520. Springer-Verlag, 1999.

J. Paredis. Steps towards co-evolutionary classification networks. In R. A. Brooks and P. Maes, editors, *Artificial Life IV, Proceedings of the fourth International Workshop on the Synthesis and Simulation of Living Systems.*, pages 359–365. MIT Press, 1994.

J. Paredis. Coevolutionary computation. *Artificial Life Journal*, 2(3), 1996.

J. Pollack and A. Blair. Coevolution in the successful learning of backgammon strategy. *Machine Learning*, 32(3): 225–240, 1998.

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

M. Potter and K. De Jong. A cooperative coevolutionary approach to function optimization. In *Proceedings from the Third Parallel Problem Solving from Nature*, pages 249–257. Springer-Verlag, 1994.

M. Potter and K. De Jong. The coevolution of antibodies for concept learning. In *Proceedings from the Fifth Parallel Problem Solving from Nature*, pages 530–539. Springer-Verlag, 1999.

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

C. Rosin. *Coevolutionary Search Among Adversaries*. PhD thesis, University of California, San Diego, 1997.

C. Rosin and R. Belew. Methods for comptetitive coevolution: Finding opponents worth beating. In L. Eshelman, editor, *Genetic Algorithms: Proceedings of the Sixth International Conference*, pages 373–380. Morgan Kaufmann, 1995.

C. Rosin and R. Belew. New methods for competetive coevolution. *Evolutionary Computation*, 5(1):1–29, 1996.

R. Salomon. Performance degradation of genetic algorithms under coordinate rotation. In L. Fogel, P. Angeline, and T. Bäck, editors, *Proceedings of the Fifth Annual Conference on Evolutionary Programming V*, pages 153–161. MIT Press, 1996.

D. Schlierkamp-Voosen and H. Mühlenbein. Strategy adaptation by competing subpopulations. In *Proceedings from the Third Parallel Problem Solving from Nature*, pages 199–108. Springer-Verlag, 1994.

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