A Sensitivity Analysis of a Cooperative Coevolutionary Algorithm Biased for Optimization

Liviu Panait, R. Paul Wiegand, and Sean Luke

George Mason University, Fairfax, VA 22030
lpanait@cs.gmu.edu, paul@tesseract.org, and sean@cs.gmu.edu

Abstract. Recent theoretical work helped explain certain optimization-related pathologies in cooperative coevolutionary algorithms (CCEAs). Such explanations have led to adopting specific and constructive strategies for improving CCEA optimization performance by biasing the algorithm toward ideal collaboration. This paper investigates how sensitivity to the degree of bias (set in advance) is affected by certain algorithmic and problem properties. We discover that the previous static biasing approach is quite sensitive to a number of problem properties, and we propose a stochastic alternative which alleviates this problem. We believe that finding appropriate biasing rates is more feasible with this new biasing technique.

1 Introduction

Coevolutionary algorithms (CEAs) are popular augmentations of traditional evolutionary algorithms (EAs). The basic elements of these augmentations lay in the adaptive nature of fitness evaluation in coevolutionary systems: individuals are assigned fitness values based on direct interactions with other individuals. Of particular interest to us are cooperative coevolutionary algorithms (CCEAs), in which interacting individuals are rewarded when they perform well together as a team, and punished when they perform poorly.

At first blush, it would seem that CCEAs may do well on large domain spaces with certain structural properties among interacting components. The intuition behind this advantage is that the algorithm searches only projections of the space at any given time, thus presenting a narrower search domain in a particular generation. Unfortunately, though CCEAs search only a projection of the problem at a time, that projection is constantly changing. The result is that it is easy for the algorithm to get tricked by misleading information provided by poor samples of the projected space. This leads to algorithms that tend to prefer strategies in one population that will do well against many strategies in the other population(s) (so-called *robust resting balance*), whether or not the strategy is globally optimal.

Understanding this particular weakness of CCEAs has led to a very obvious mechanism for correcting it: bias the algorithm to seek ideal collaborations rather

K. Deb et al. (Eds.): GECCO 2004, LNCS 3102, pp. 573–584, 2004.

than robust resting balance. Early investigation into this idea has proved fruitful [1], but this study left at least two things unexplored. First, applying any kind of bias toward ideal collaboration would require learning that bias as the run proceeds, and it is unclear how best to do this. Second and specifically pertinent to this paper, it is not clear how sensitive the degree of bias is to various settings. Large amounts of bias may be unwise if ideal-partnership estimates are poor. But depending on problem sensitivity, small amounts of bias may have almost no effect. A smooth, relatively insensitive biasing procedure would then be a useful part of a successful cooperative coevolution technique.

The paper continues with a brief introduction to cooperative coevolution and to pathologies that may prevent it from achieving optimal results, followed by a description of our previous approach to improving coevolutionary search. We then perform a sensitivity study on a simplified version of the method, and show that results are greatly affected by problem domain properties. Next, we propose an alternative approach to biasing the coevolutionary search, and show that it is less sensitive to algorithm and problem features.

2 Background

In the Potter model of cooperative coevolution [2], which we use in this paper, 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. To combat noise in the evaluation process due to choice of partners, multiple evaluations are usually performed. An individual's fitness could be the maximum over such evaluations, or the mean, among other approaches [3]. Aside from evaluation, the populations are evolved independently. Applications of this method include optimization of inventory control systems [4], learning constructive neural networks [5], and rule learning [6,7].

Suppose we are optimizing a three argument function $f(x, y, z)$. One might assign individuals in the first population to represent the x argument, the second to represent y, and the third to represent z. 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 populations (e.g., y and z) in order to obtain an objective function value with a complete solution, $f(x, y, z)$. A simple example collaboration method is to choose representing members by using the most fit individual from those populations as determined by the previous round of evaluations. Another approach is to pick partners at random from the other population. Once a complete solution is formed, it can be evaluated and the resulting score can be assigned to the individual.

Unfortunately, as is often the case for coevolution, complications in the dynamical behaviors of these algorithms often arise in application. The algorithms frequently perform poorly on what seem to be relatively simple problems, falling into wells of mediocrity or collapsing prematurely due to sudden asymmetric

losses in population diversities [8,9]. These problems have led researchers to focus on more sophisticated collaboration methods to attempt to address these pathologies [10,11,12].

Recent formal and empirical analysis of the CCEA has suggested a pathology special to CCEAs: *relative overgeneralization* [8]. This research states that cooperative coevolutionary systems are more attracted to points of robust resting balance than to ideal collaboration. From a game-theoretic viewpoint, this tendency can be described as an attraction to the Nash equilibria with the highest joint reward distributions, and not necessarily to those equilibria associated with the global optimum. In short, existing cooperative coevolutionary algorithms are not necessarily well-suited for static, single-objective optimization. Though no real analytical work exists for CCEAs on dynamic optimization problems, the same pathology will very likely confound CCEAs on these tasks, as well.

One straightforward approach to dealing with some pathologies is to bias the CCEA toward searching for ideal collaborations. This idea has been posited and explored at cursory level, with very encouraging results [1]. This existing study identifies bias rate sensitivity as a concern, but does not explore it.

In this paper we will perform this biasing by using the *actual* ideal collaborator for a given individual: in the problem space being tested we can compute this collaborator trivially. Of course, for most real-world problems this is hardly the case, and in the worst case the actual ideal collaborator can only be found via a full search over the individual's collaboration space. Thus in most cases one would need some approximation function for collaborators. In future work we hope to examine a variety of methods for approximating the ideal collaborator: for example, keeping histories of previous collaboration success, or performing small hill-climbing searches for a collaborator. But for purposes of this paper we concern ourselves with the theoretical bound: even *if* you knew the ideal collaborator, what problems in bias sensitivity still arise, and how might one go about solving them?

3 Biasing the CCEA

The biasing method employed here is relatively simple: assess the objective function value is a weighted sum of two terms. The first term is the immediate reward received when the individual component is teamed with partners from other populations chosen using some collaboration scheme. The second term is the reward that the individual would have received had it interacted with its ideal collaborators[1].

Our focus in this paper is on the fraction of reward due to each term and the factors influencing how this fraction is determined. Panait *et al* [1] propose

[1] Fitness for the individual involves teaming the individual with one or more partners, computing the objective function value in each case, and assigning a single fitness score in the manner stipulated by the collaboration scheme. The combination of the biased and interactive pieces of the objective function is orthogonal to the issue of combining objective function values when using multiple collaborators

a fitness assessment method that is a linear, weighted sum of these two components. Equation 1 below describes this idea mathematically. Here the fitness of argument x_a is being assessed by combining the result of the objective function, g, with an estimation of an ideal projection of that function, g'_a. The parameter δ controls the degree of emphasis the fitness places on the estimate of the ideal.

$$f(x_1, \ldots, x_a, \ldots, x_k) = (1 - \delta) \cdot g(x_1, \ldots, x_a, \ldots, x_k) + \delta \cdot g'_a(x_a) \qquad (1)$$

The intuitive high-level picture for this tradeoff is clear. At one extreme, when $\delta = 1$, the algorithm will trust only the estimate, and the states of the other populations are entirely irrelevant. It is no longer a coevolutionary system at all; it is k EAs searching the k-projected component spaces independently in parallel. This would be ideal if the estimate were completely trustworthy, though this will not typically be the case. At the other extreme, $\delta = 0$, the algorithm trusts only the collaborative assessment of the objective function. This is the traditional CCEA, with all the challenges that system faces.

Unfortunately, finding a good setting of δ is not an easy task. First of all, the coevolutionary algorithm may have very poor estimates for the ideal partnership. The result of high values of δ and such poor estimates could result in significantly reduced performance. Second, even supposing that such estimates are appropriately learned, it seems intuitive that different levels of δ may be appropriate at different stages in the coevolutionary search. This may be accomplished by either predefined or adaptive methods for setting it. Initially, the biased coevolutionary system does not have much information about the search space. As δ reflects the confidence in the accuracy of the estimated ideal partners, its initial values may be set relatively low. However, as the search progresses, these estimations get more and more accurate, and δ might be adaptively adjusted to higher values to help guide the search process to promising areas.

In this paper we use a simple, static value for δ throughout a run and focus our attention on how different static values affect final run-time performance. Ideally, we would like changes in δ to result in a smooth change in performance. Sadly, this is not guaranteed. The problem is that the effect of changing δ can be highly sensitive to domain or other parameters in a nonlinear fashion. When the transition in performance is very smooth and gradual, it should be easier to find appropriate settings; however, when the transition is sharp and sudden, setting δ appropriately may be quite difficult.

4 Problem and Algorithm Properties

We construct a class of problem domains designed to illustrate certain salient properties of the EC algorithm: the *Maximum of Two Quadratics* (or MTQ) can offer a range from simple to very difficult problem instances. Such problems instances can resolve or exacerbate the relative overgeneralization pathology of cooperative coevolution by adjusting the relative contributions of joint rewards implicitly [8]. It does this by providing a class of maximization problems with two peaks defined as

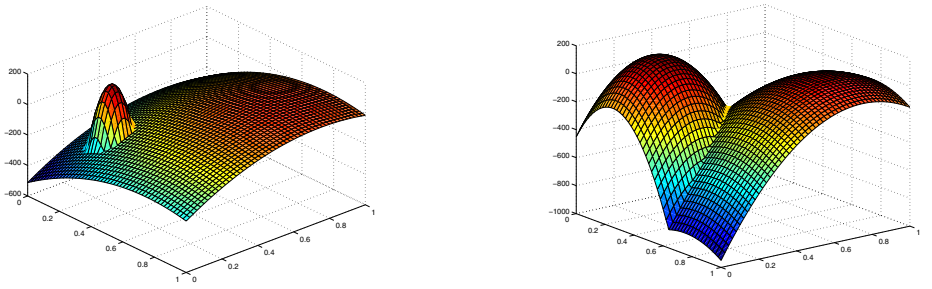

Fig. 1. Example *Maximum of Two Quadratics* problems, illustrating $(S_1 = 1.6, S_2 = 0.5)$, and $(S_1 = 0.5, S_2 = 0.5)$.

$$MTQ(x, y) \leftarrow \max \begin{cases} H_1 * (1 - \frac{16*(x-X_1)^2}{S_1} - \frac{16*(y-Y_1)^2}{S_1}) \\ H_2 * (1 - \frac{16*(x-X_2)^2}{S_2} - \frac{16*(y-Y_2)^2}{S_2}) \end{cases} \quad (2)$$

where x and y take values ranging between 0 and 1. Figure 1 illustrates some example MTQ problem instances. Different settings for H_1, H_2, X_1, Y_1, X_2, Y_2, S_1 and S_2 affect the difficulty of the problem domain in one of the following aspects:

Peak height: H_1 and H_2 affect the heights of the two peaks. Higher peaks may increase the chances that the algorithm converges there.

Peak coverage: S_1 and S_2 affect the area that the two peaks cover: higher values for one of them result in a wider coverage of the specific peak. This makes it more probable that the coevolutionary search algorithm will converge to this peak, even though it may be suboptimal.

Peak relatedness: Different values for X_1, Y_1, X_2 and Y_2 result in changes in the locations of the centers of the two quadratics, which also affect the relatedness of the two peaks: similar values of the x or y coordinates for the two centers imply higher overlaps of the projections along one or both axes.

Aside from the impact of the problem domain properties, our sensitivity study targets three algorithmic settings:

Biasing rate: Altering the biasing rate allows us to study the result of CCEA runs at various fixed rates of $\delta \in [0, 1]$. We are interested in how performance degrades during this transition.

Population size: The population size affects how much the coevolutionary algorithm samples the search space. It is easy to show that coevolution using a bias rate δ of 1.0, combined with infinite populations and perfect knowledge of maximal projections, will converge to the unique optimum with probability 1. We expect similar results for large populations as well.

Collaboration scheme: The CCEA algorithm tries to simplify the search process by decomposing the candidate solutions into components and coevolving them in separate populations. The only information such a population can get about the overall progress of the search process is through collaborators – samples usually representative of the status of the other populations. For some spaces, an increased number of collaborators may better capture the intricacies of the search space [10,12].

5 Bias Sensitivity

All experiments used the MTQ class of problems. The coevolutionary search process used two populations, one for each of the variables. Each such population used a real-valued representation, with individuals constrained to values between 0 and 1 inclusive. Non-adaptive Gaussian mutation (mean 0 and standard deviation 0.05) was the only variational operator. Each population used tournament selection of size 2, and the best individual survived automatically to the next generation. The search lasted for 50 generations, after which time the best individuals in each population were at, or very near, one of the two peaks. Each point in Figures 2-5 is computed over 250 independent runs. All experiments were performed with the ECJ system [13].

Unless stated otherwise, each population consisted of 32 individuals. The default collaboration scheme used two collaborators from each population: the best individual in the previous generation was always selected, and the other individual was chosen at random. The objective component of an individual's fitness was assessed as the better of its results when teamed with each of the collaborators. The default values of the parameters for the first (suboptimal) peak were H_1=50, X_1=$\frac{1}{4}$, Y_1=$\frac{1}{4}$, and S_1=1.6. The second (optimal) peak was characterized by H_2=150, X_2=$\frac{3}{4}$, Y_2=$\frac{3}{4}$, and S_2=$\frac{1}{32}$. With these settings, the two peaks are near at opposite corners of the search space.

5.1 Biasing and Domain Features

The first set of experiments investigated the relationship between the biasing rate and the three problem domain features previously described: the relative heights, coverages and locations of the peaks. There are 11 experimental groups for each property, one for each value of $\delta \in [0, 1]$ in increments of 0.1. Figure 2 shows the mean final results of these 33 groups.

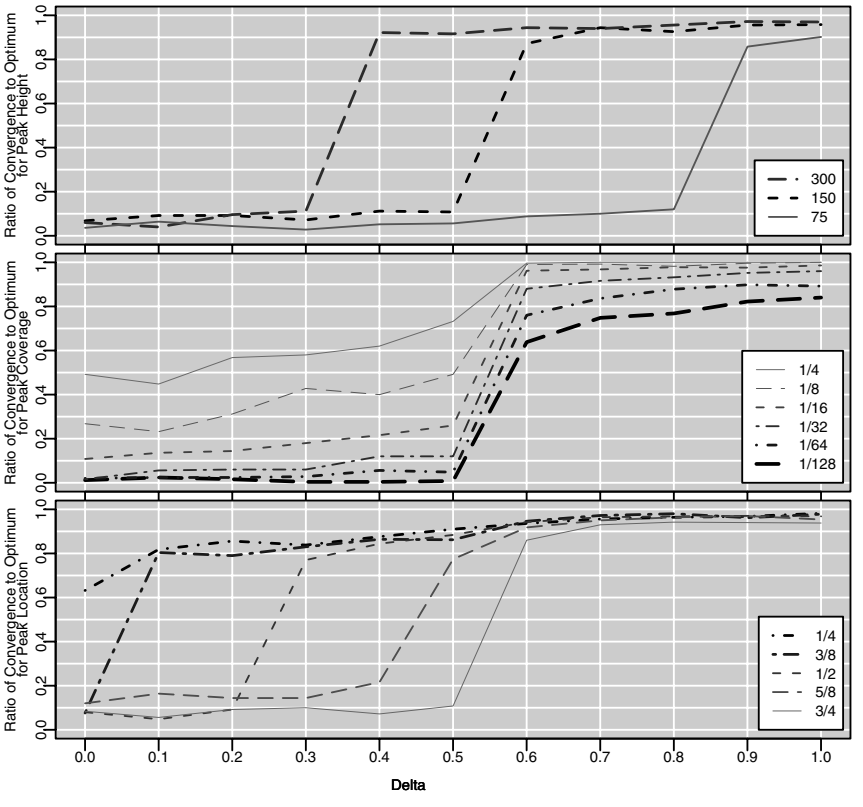

Fig. 2. Convergence ratios for peak height (top), peak coverage (center) and peak relatedness (bottom). x axis shows biasing rate δ, and y axis shows ratio of the 250 trials that converged to, or very near, the global optimum.

Peak height: We kept H_1 constant at 50, and set H_2 to 75, 150 and 300. The results indicate that less than 10% of runs converge optimally when the rate of biasing is low, while the ratio increases to more than 90% when using high biasing rates. Unfortunately, there is no smooth transition between these two extremes: rather, small modifications to the biasing rate may change the rate of convergence to optimum by as much as 70–80%. Moreover, the relative difference in peak height directly affects where these sudden jumps in performance appear. This suggests that δ may not only be quite sensitive to relative differences in peak height in the sense that small changes may have radical effects, but it may even be quite difficult to find the transition range of δ values at all.

Peak coverage: S_2 was set to $\frac{1}{128}$, $\frac{1}{64}$, $\frac{1}{32}$, $\frac{1}{16}$, $\frac{1}{8}$, and $\frac{1}{4}$, while S_1 was constantly 1.6. Here, the location of transition is more consistent among the various values, but the transitions themselves are still relatively abrupt. It also appears that the relative peak coverages cause more variation in results when the bias rate is small,

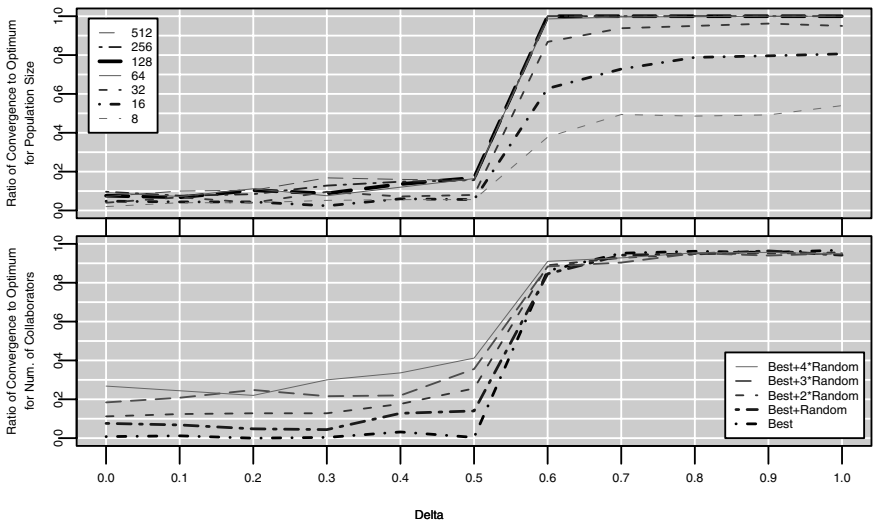

Fig. 3. Convergence ratios for population size (top) and collaboration scheme (bottom). x axis shows biasing rate δ, and y axis shows ratio of the 250 trials that converged to, or very near, the global optimum.

while the curves at the other extreme of the graph appear close together. The results indicate that peak coverage may alleviate the sensitivity of the algorithm to the δ parameter: the wider the peak, the more gradual the transition when varying the bias rate.

Peak relatedness: The bottom graph in Figure 2 shows the ratio of runs converged to optimum when varying the relative location of the two peaks: the Y_2 parameter was set to $\frac{1}{4}$, $\frac{3}{8}$, $\frac{1}{2}$, $\frac{5}{8}$ and $\frac{3}{4}$. These settings gradually transitioned the relative peak positions from completely opposite locations to ones aligned along one axis. Similar to peak height, the peak relatedness had a significant effect on the ratio of runs converged to optimum: the more related the peaks, the less biasing is required to assure good performance. However, the curves have an abrupt transition between lower and higher rates of convergence to optimum. Moreover, the location of this transition depends on the actual degree of peak relatedness, which suggests that δ may be highly sensitive to this parameter.

5.2 Biasing and Algorithm Features

A second set of experiments investigated the relationship between the biasing rate and the two other algorithm features previously described: the population size and the collaboration scheme. Again, there are 11 groups for each of these two parameters corresponding to each of the δ settings. The results are presented in Figure 3.

Population size: We set the size of each of the two populations to 8, 16, 32, 64, 128, 256 and 512. As expected, extremely small populations have a smaller chance of reaching the optimum even for high biasing rates. We observe the same abrupt shift in performance as we saw in the previous experiments. The results suggest that increasing the population size does not necessarily alleviate δ sensitivity.

Collaboration scheme: The default setting in all previous experiments used two collaborators to evaluate the fitness of each individual: the best performing individual from the other population in the previous generation, and also a random individual. To test sensitivity to this collaboration scheme, we varied the number of random individuals from 0 to 4; the best individual from each population in the previous generation was always used. The bottom graph in Figure 3 shows that the collaboration scheme has some influence over the performance of the algorithm at low biasing rates, but it has no effect when higher biasing rates are used. Again, the abrupt change in performance indicates that the δ parameter can be highly sensitive, regardless of the collaboration methodology.

6 A Probabilistic Biasing Alternative

In the previous section, we learned that the degree of bias rate, δ, can be very sensitive to certain problem properties, such as differences in relative peak height, while being less sensitive for others, such as peak coverage. Unfortunately, we also learned that certain coevolutionary algorithmic parameters, such as the size of the populations and the number of collaborators used during evaluation, may not be very helpful in making the task of setting δ more feasible. What can be done at the algorithm level to make δ more flexible?

To uncover a possible alternative, let's look again at the results of the peak height experiments. Recall that for larger differences in peak heights, a wider range of biasing rates results in a high ratio of convergence; however, when one peak is only slightly higher than the other, the range of high convergence ratios is much smaller. The transition is abrupt, but the location of the transition shifts depending on the peak height differences. Why might this be so?

Our hypothesis is that this extreme sensitivity of the biasing method to the relative peak heights is caused by the linear combination of the two fitness components: the fitness when teamed with collaborators, and the one when in combination with the ideal teammates. The higher the optimal peak, the lower the bias rate it needs to dominate the other term. However, if one peak is slightly higher than the other, the algorithm requires more biasing to locate the optimum.

We propose a new alternative biased coevolutionary algorithm to deal with this issue. The new algorithm does not combine the two components into a unique fitness. Rather, each individual is assigned two fitnesses: the objective one when combined with the collaborators from other populations, and another one indicating the performance of the individual when in combination with its ideal teammates. When comparing two individuals, with probability δ we will compare based on the first "fitness"; else we will compare based on the second.

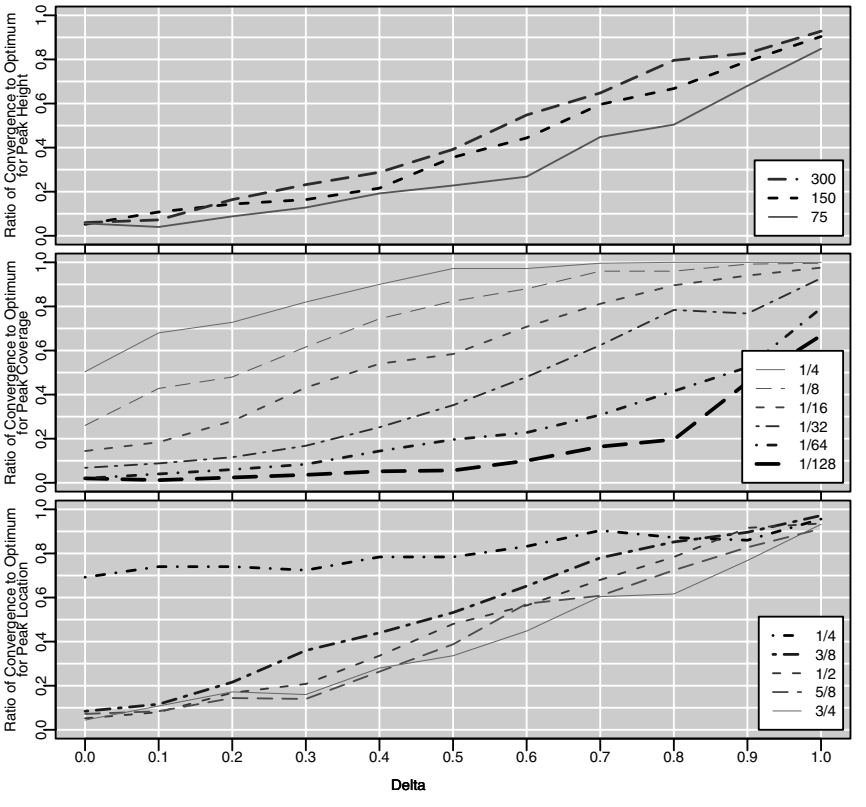

Fig. 4. Convergence ratios for probabilistic biasing when varying peak height (top), peak coverage (center) and peak relatedness (bottom). x axis shows biasing rate δ, and y axis shows ratio of the 250 trials that converged to, or very near, the global optimum.

We performed the same sensitivity analysis for the new algorithm. The results are presented in Figures 4 and 5. In all cases, the new algorithm does not exhibit the sudden jumps in performance that the original one did. This suggests that it is an improved algorithm that is significantly less sensitive to the settings we have investigated.

7 Conclusions and Future Work

Cooperative coevolution remains a powerful tool for a wide variety of problems. Unfortunately, it is fraught with difficulties, and understanding the reasons for these difficulties may help to improve CCEA algorithms for optimization. Biasing the CCEA toward optimal collaboration appears to be just such an improvement: it directly addresses one of the fundamental dynamical problems with cooperative coevolution. Still, it is important to understand how this augmenta-

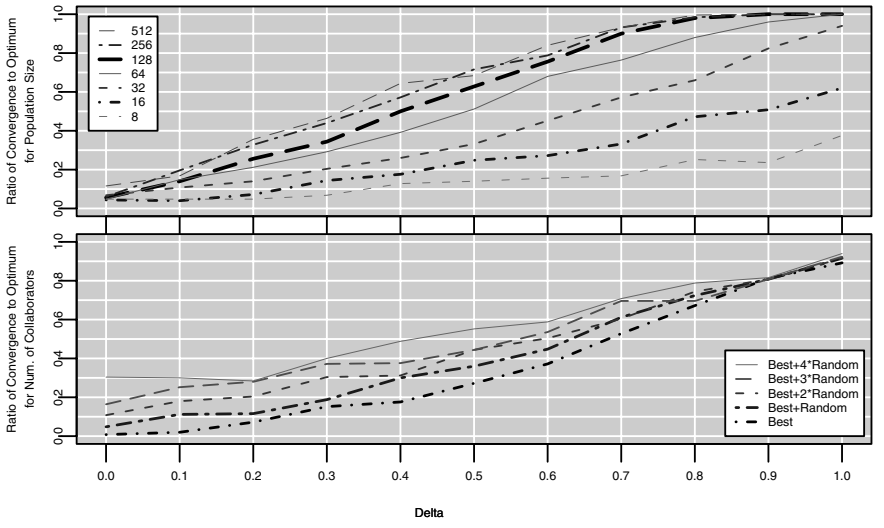

Fig. 5. Convergence ratios for probabilistic biasing when varying population size (top) and collaboration scheme (bottom). x axis shows biasing rate δ, and y axis shows ratio of the 250 trials that converged to, or very near, the global optimum.

tion works—what its benefits and drawbacks are. In particular, since the biasing method relies on a parameter that specifies the degree to which the algorithm trusts the bias over the collaborative interactions, understanding the nature and impact of this parameter is of paramount concern.

This paper takes a sizable step towards understanding how this δ parameter influences the biased CCEA by investigating its sensitivity. Starting with a model that combines traditional collaborative interaction and an optimization bias, we assume that the maximal projection of the problem is known and use this for the bias. The MTQ problem class helps elicit information about the relationship between certain problem properties and algorithmic parameters. Using this problem, we discovered that applying δ as a means to control the linear combination of the two measures results in high sensitivity to a variety of problem parameters. For two of those parameters, the location of δ sensitivity varied widely. This makes setting or adjusting δ a real challenge.

We offer a probabilistic alternative to the linear combination of bias and collaborative fitness assessment that seems to allow for more flexibility in the δ parameter. The result is an improved method that is much less sensitive to the minor differences in the degree of bias.

Our next step is to investigate tractable methods for learning estimations of ideal partners. In addition, we would like to study how δ might be dynamically or adaptively altered during the run to help CCEAs search more efficiently. Our hope is that a biased CCEA will help focus the power of cooperative coevolution more appropriately for optimization tasks.

References

1. Panait, L., Wiegand, R.P., Luke, S.: (Improving coevolutionary search for optimal multiagent behaviors) 653–658
2. Potter, M.: The Design and Analysis of a Computational Model of Cooperative CoEvolution. PhD thesis, George Mason University, Fairfax, Virginia (1997)
3. Wiegand, R.P., De Jong, K.A., Liles, W.C.: (The effects of representational bias on collaboration methods in cooperative coevolution) 257–268
4. Eriksson, R., Olsson, B.: Cooperative coevolution in inventory control optimisation. In Smith, G., Steele, N., Albrecht, R., eds.: Proceedings of the Third International Conference on Artificial Neural Networks and Genetic Algorithms, University of East Anglia, Norwich, UK, Springer (1997)
5. Potter, M., De Jong, K.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evolutionary Computation **8** (2000) 1–29
6. Potter, M., De Jong, K.: (The coevolution of antibodies for concept learning) 530–539
7. Potter, M.A., De Jong, K.A., Grefenstette, J.J.: A coevolutionary approach to learning sequential decision rules. In: Proceedings from the Sixth International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, Inc. (1995) 366–372
8. Wiegand, R.P.: An Analysis of Cooperative Coevolutionary Algorithms. PhD thesis, George Mason University, Fairfax, Virginia (2004)
9. Ficici, S., Pollack, J.: Challenges in coevolutionary learning: Arms–race dynamics, open–endedness, and mediocre stable states. In et al, A., ed.: Proceedings of the Sixth International Conference on Artificial Life, Cambridge, MA, MIT Press (1998) 238–247
10. Wiegand, R.P., Liles, W., De Jong, K.: (An empirical analysis of collaboration methods in cooperative coevolutionary algorithms) 1235–1242
11. Bull, L.: On coevolutionary genetic algorithms. Soft Computing **5** (2001) 201–207
12. Bull, L.: (Evolutionary computing in multi-agent environments: Partners) 370–377
13. Luke, S. ECJ 9: A Java EC research system. http://www.cs.umd.edu/projects/plus/ec/ecj/ (2002)