Applying Diffusion to a Cooperative Coevolutionary Model

R. Paul Wiegand

paul@tesseract.org
Computer Science Department
University of North Carolina – Charlotte, NC 28223, USA

Abstract. Perhaps one the newest and of the more interesting cooperative approaches to evolutionary computation which has been more recently explored is the area of mutualism. In mutualistic methods, the problem is subdivided into modular components where each component evolves separately, but is evaluated in terms of the other components. In this way the problem may be cooperatively solved. In an attempt to give this novel approach more adaptive ability, in this paper we explore the effects of adding varying degrees of population diffusion via a mutatable tagging scheme applied to individual chromosomes.

1. Introduction

Despite the tremendous scientific advancements of this century, and throughout history, computer science still remains inordinately inferior to nature in its ability to solve complex problems. Indeed, much advancement in the various fields of the science has come from borrowing solving techniques from nature itself. Surely there can be no mistaking this in the area of *Genetic Algorithms* (GA), arguably pioneered by John Holland (Goldberg, 1989; Holland, 1992) in 1975, as well as the more general area of *Evolutionary Computation* (EC), born as early as the late 1950's/early 1960's by a men such as Bremermann, Friedberg, Box, and Friedman (Fogel, to appear).

Borrowing from the concepts of Darwinian evolution, these early pioneers realized that the precepts of emergence and natural selection were well suited for many types of hard problems faced by computer scientists. They developed techniques for creating populations of individuals, consisting of potential solutions encoded in chromosome-like forms, and applying genetic operators (such as mutation and crossover) as these population members interbreed and are selected based on fitness values determined using varying evaluation methods.

Nature continues to be a source of inspiration for the field of EC. Concepts of cooperation and competition have since been applied in many forms, as have other forms of subpopulation and speciation techniques. Moreover, using multiple populations as a means to accomplish both coevolutionary solving as well as solving simul-

taneously for multiple near optimum solutions has captivated the attention of many EC researchers.

Further, recent coevolutionary advancements have included the concept of *mutualism* (Potter, 1997). Here problems are broken up such that a given subpopulation consists of individuals representing a portion of a solution. These subpopulations are evolved entirely separately, with the exception of evaluation, during which representatives are used in order to produce fitness values via cooperation.

In this study, we explore combining techniques used in studies regarding multiple subpopulation research, including mutualism. This paper discusses the use of a mutatable tagging system to allow individuals in a mutualistic model to move from subpopulation to subpopulation via a technique called *diffusion* (Spears, 1994). We will discuss this background research in more detail in section 2. In section 3, we will describe how these historic techniques were uniquely combined for our study. The results are described in section 4, and the conclusion is discussed in the final section.

2. Background

Using multiple subpopulations in a GA is a technique that has been harnessed for many purposes by many researches in the field of EC. Goldberg and Richardson's (1987) work showed how a simple GA can be used to find multiple optimum and suboptimum peaks during function optimization by using two techniques, *restricted mating* and *sharing*. Here individuals were only allowed to breed with other members of the population that were genetically similar, implicitly creating different species within the total ecology of a single GA. Sharing was then employed to prevent *crowding* of individuals on the tallest peaks by modifying fitness values of individuals in the subpopulations dynamically.

William Spears continued this research (Spears, 1994), modifying it for simplicity and performance by providing, more explicitly, the use of multiple subpopulations within the greater population. Further, individuals were restricted in mating to those individuals which were likewise a member of the same subpopulation. Membership to a given subpopulation was identified by the use of *tags*, additional bits affixed to the chromosome which specified to which species that individual belonged. Moreover, fitness sharing was implemented by taking the fitness value as formed by the evaluation process of a given individual, and dividing it by the total number of individuals which were also members of the same subpopulation. Subpopulations would thus grow and shrink dynamically from generation to generation depending on selection. We expect larger peaks to have larger subpopulation sizes, and smaller peaks to have their proportion, as the GA becomes more stable during convergence.

Although Spears performed his experiments in this study without mutation of the tag bits, he briefly discusses the concept of *diffusion*, in which the tag bits can be mutated at the same or a differing rate as the other bits in the chromosome. This rate is known as the *rate of diffusion.* In this way, parents from one generation may yield a mutated child, such that the offspring is a member of a different species.

In addition to multiple species being used in this way, it is also possible to use speciation as a means of accomplishing coevolution. Although the vast majority of such

research has been primarily focussed in the area of competitive species (Schlierkamp-Voosen et al., 1994; Cohoon et al., 1987; and Tanese, 1989), there have been some significant studies of cooperative systems. Indeed, Mitchell Potter and Kenneth De Jong use this very idea in function optimization (Potter et al., 1994), yielding the concept of *mutualism*. In mutualism, a given problem is broken down into components, where each is evolved mutually, but in isolation to the other components. During evaluation of an individual from a given subpopulation, representatives are selected from the other subpopulations in order to form a complete solution, and fitness is assigned by using this cooperative structure. Potter and De Jong refer to these systems as *cooperative coevolutionary genetic algorithms* (CCGAs).

Although Potter investigates more dynamic methods of creating and eliminating species for other types of problems (Potter, 1997), such as structural construction of neural networks, what is of more significance to us is the work with function optimization, which has a more static nature in terms of component construction. When optimizing functions, Potter assigns each species to a given variable of a multivariable function. The populations are homogeneous in the sense that they are all represented in the same bit string structure, and the same genetic operators are applied (albeit separately) to each population, though this need not be the case for all problems, as he clearly discusses (Potter, 1997).

Selecting a representative, or a *collaborator*, can be done in many ways, but most easily is accomplished by selecting the member with the highest fitness value from each of the collaborating subpopulations[1]. So each generation, all the individuals in a species under evaluation will be applied to the objective function one at a time, using the most fit individuals in the previous generation from the other species to form a complete solution. In this way these collaborators *cooperate* with the subpopulation currently being evaluated. Each species is evaluated in turn in this manner in a full *ecosystem generation*[2]. A good comparison of a standard GA to the CCGA in pseudocode was provided by Potter (1994) and is shown below in figures 1 and 2.

```
gen = 0
Pop(gen) = randomly initialized population
evaluate fitness of each individual in Pop(gen)
while termination condition = false do begin
        gen = gen + 1
        select Pop(gen ) from Pop(gen - 1) based on fitness
        apply genetic operators to Pop(gen)
        evaluate fitness of each individual in Pop(gen)
        end
```

Fig. 1. The structure of a traditional GA

[1] In the case of the generation 0, we randomly select collaborators.

[2] Potter (1997) defines generation "…to be a complete pass through the select, recombine, evaluate, and replace cycle of a single species…" and an ecosystem generation to be "…an evolutionary cycle through all species being coevolved."

```
gen = 0
for each species s do begin
        Pop_s(gen) = randomly initialized population
        evaluate fitness of each individual in Pop_s(gen)
        end
while termination condition = false do begin
        gen = gen + 1
        for each species s do begin
                select Pop_s(gen ) from Pop_s(gen - 1) based on fitness
                apply genetic operators to Pop_s(gen)
                evaluate fitness of each individual in Pop_s(gen)
                end
        end
```

Fig. 2. The structure of a Cooperative Coevolutionary GA

Notice that there will be more evaluations per ecosystem generation in the CCGA than in a traditional GA generation. In fact, the number of evaluations is *n* * *population size*, where *n* is the number of species (or arguments of the function). It is difficult to draw a comparison between generations in a traditional GA, and ecosystem generations in a CCGA for this reason. More appropriately, we look at the total number of evaluations performed. It is clear, however, that for the same number of function evaluations, the GA will complete more generations than the CCGA will complete ecosystem generations.

This novel approach gives us an idea how a problem can be statically sub-divided and solved in a coevolutionary, mutualistic manner. In the next section we will discuss how we were able to provide a bit more flexibility in allowing the CCGA to "tune" subpopulation sizes to a particular problem landscape, as well as providing possible future research possibilities with regards to more natural mechanisms for species to arise and become extinct.

3. Tagging and Mutualism

The application of diffusion to the CCGA model represents a very natural marriage of Potter's mutualistic model and Spear's tagging scheme. However, despite the complementary relationship of these techniques, there still remain several issues raised by the technique, which must be addressed.

Indeed, these concepts, as we have already shown, are not substantively dissimilar, as both models make use of the concept of multiple subpopulations, albeit with differing representations. As with Spears tagging, our model, here called *a diffusable cooperative coevolutionary genetic algorithm* (DCCGA), segregates individuals into specific subpopulations by means of a binary tag in the chromosome. These tags are

subject to mutation at a differing rate than the other bits (the rate of diffusion), allowing parents of one generation to yield offspring that belong to a different species.

The DCCGA model, however, still subdivides the problem into cooperating populations, each representative of arguments in the function. Like Potter's model, collaborators are chosen as representatives for evaluation, and each subpopulation is evolved mutually.

The application of such a tagging scheme allows for the possibility of more species than arguments for the function; though, since the number of possible distinct species (S_o) is given as follows:

$$S_o = 2^n,$$

where *n* is the number of bits in the tag.

With this observation, it becomes imperative that we draw a distinction between *niche* and *species*. In the DCCGA, a *niche* is an ecological space where individuals from that space represent encoded values for a particular argument of a given function. On the other hand, a *species* is a population where the individuals of that population can interbreed, but cannot breed outside of that population. In the case of the DCCGA, there may be more than one species that fill a particular niche. Indeed, more often than not, there will be several occurrences of 2 species which representative of the same niche, i.e. two different tags represent the same parameter.

In our case, the solution is to construct a table to map each species tag to an appropriate niche. Such a mapping necessitates that there be some degree of overlap. The table is constructed sequentially, by consecutively assigning each tag to a function argument, overlapping when necessary. For example, in the case of a function which requires 5 variables, the smallest S_o which can be used is 8 (3 bits, $2^3 = 8$). As such, the following table can be constructed:

Tag	**Function Argument**
000	x_1
001	x_2
010	x_3
011	x_4
100	x_5
101	x_1
110	x_2
111	x_3

Tab. 1. Table illustrating the mapping of species to niche for a 5 variable function

We can see from this table that, in three cases (x_1, x_2, and x_3) there will be two different species that contain individuals representing potential values for each variable.

The next issue is that of how a single collaborator can be chosen between the two populations. In our DCCGA, this is done by simply selecting the collaborator from the subpopulation with the highest average fitness. Such a resolution is by no means

the only solution to this issue, nor the most sophisticated, but should suffice for the general case.

Another issue that arises from the application of diffusion to Potter's model is that of the potential for extinction. Since the tags are subject to mutation, the population sizes are dynamic. While this is, in part the intended goal of this addition, it also serves to bring about a condition from which no evaluation is possible: extinction of all the species covering a specific niche. To resolve this, a certain degree of preservation is necessary. Here we choose to preserve the best 10% of each subpopulation. This alleviates this issue by establishing an implicit minimum population size for each species, although presents the possibility of a loss of diversity in species which have a depleted population size due to diffusion.

There are several advantages to this technique. First of all, subpopulation sizes are dynamic, consequently the search may favor one argument over another. Also, diffusion brings the gene pool of a population possible diversity benefits, in particular for asymmetric functions. In addition, the overall process is more natural, allowing for the possibility of an individual being born genetically dissimilar to its parents such that it is no longer of the same species. This indicates the possibility of future research using similar methods to provide more natural mechanisms of species creation and extinction in the dynamic CCGA approaches discussed by Potter in his dissertation (Potter, 1997). Some ideas to this end are discussed in the conclusion of this paper.

4. Experimental Results

In order to provide a thorough study, each of 12 experimental groups were used to optimize a total of 5 different functions. The groups were constructed in order to compare results of a standard GA, both with and without tagging (SSGA), CCGA, and DCCGA techniques. To get a clearer picture regarding the effect of diffusion, the diffusion rate is varied. The groups are shown below in table 2, on the following page.

With the obvious exception of those parameters listed in table 2, the parameters chosen remained consistent for all runs of all groups. These are as follows:

Representation:	binary (16 bits per function variable)
Selection:	fitness proportionate with linear scaling
Mutation:	bit-flip, $p_m=1/L$ (where L is the length of the chromosome)
Crossover:	two-point, $p_c=0.60$
Population:	100 individuals per population
Termination:	100,000 function evaluations

The functions chosen to optimize, as well as the above parameters, were chosen primarily for similarity of Potter's study. Indeed the functions optimized in this study, listed in the table 3 (also found on the following page), represent the very same functions that Potter and De Jong used in their 1994 study of CCGAs.

Experimental Group	Description of Group	Elite Preserved	Rate of diffusion
GAv1	Standard elite genetic algorithm	1	0.00
GAv2	Elite genetic algorithm	10	0.00
CCGAv1	Potter's cooperative coevolutionary genetic algorithm	1	0.00
CCGAv2	Potter's cooperative coevolutionary genetic algorithm	10	0.00
DCCGA0	Diffusable genetic algorithm	10	0.00
DCCGA2	Diffusable genetic algorithm	10	0.02
DCCGA5	Diffusable genetic algorithm	10	0.05
DCCGA10	Diffusable genetic algorithm	10	0.10
SSGA0	Simple subpopulation GA (Spears tagging)	10	0.00
SSGA2	Simple subpopulation GA (Spears tagging)	10	0.02
SSGA5	Simple subpopulation GA (Spears tagging)	10	0.05
SSGA10	Simple subpopulation GA (Spears tagging)	10	0.10

Tab. 2. Table describing the groups used in the experiment.

Definition	Range & Values
$f_1(\vec{x}) = -c_1 \cdot \exp\left(-c_2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(c_3 x_i)\right) + c_1 + e$	-30.0..30.0 $n = 30$ $c_1 = 20$ $c_2 = 0.2$ $c_3 = 2\pi$
$f_2(\vec{x}) = 1 + \sum_{i=1}^{n}\frac{x_i^2}{4000} - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	-600.0..600.0 $n = 10$
$f_3(\vec{x}) = 418.9829n + \sum_{i=1}^{n} x_i \cdot \sin(\sqrt{\lvert x_i \rvert})$	-500.0..500.0 $n = 10$
$f_4(\vec{x}) = 3.0n + \sum_{i=1}^{n} x_i^2 - 3.0 \cdot \cos(2\pi x_i)$	-5.12..5.12 $n = 20$
$f_5(\vec{x}) = 100(x_1^2 - x_2^2)^2 + (1 - x_1^2)^2$	-2.048..2.048

Tab. 3. Table describing the functions that were optimized in the experiment.

In all cases the true global minimum of the function is zero. In all but two cases, this minimum is found at the point $\vec{x}$ = (0,0, ···). For the function f_3, the minimum value of zero can be found at the point $\vec{x}$ = (420.9687, 420.9687, ···); however, in the case of f_5, the zero value is found at point $\vec{x}$ = (1,1).

The results for solution value are shown in table 4 below. Each group was run for each function 20 times. The results reported below indicate the average of the "best-ever" values found in the 100,000 evaluations of each trial.

	f_1	f_2	f_3	f_4	f_5
GAv1	1.59E+01	8.89E+00	2.91E+01	4.16E+02	3.63E-03
GAv2	1.24E+01	*8.39E-02*	1.15E+01	4.23E+01	*5.38E-05*
CCGAv1	1.7E+00	6.05E-01	*2.91E-01*	3.86E-01	1.58E-04
CCGAv2	*3.89E-01*	2.18E-01	3.89E+00	4.47E-01	2.79E-04
DCCGA0	9.15E-01	5.21E-01	4.93E+00	3.38E-01	6.08E-04
DCCGA2	4.86E-01	5.27E-01	6.75E+00	3.74E-01	3.35E-04
DCCGA5	4.71E-01	6.11E-01	3.70E+00	2.35E-01	1.86E-04
DCCGA10	5.19E-01	6.43E-01	5.79E+00	*1.41E-01*	3.40E-04
SSGA0	1.99E+01	7.42E+00	3.84E+01	8.67E+02	2.13E-02
SSGA2	2.03E+01	1.05E+01	1.05E+02	1.83E+03	3.31E-02
SSGA5	2.03E+01	1.05E+01	1.25E+02	1.99E+03	3.84E-02
SSGA10	2.02E+01	2.76E+00	1.33E+02	2.08E+03	1.21E-02

Tab. 4. Table displaying the solution quality results for all 5 functions and 12 experimental groups. These values represent averages across 20 trials.

	f_1	f_2	f_3	f_4	f_5
GAv1	2.67E+00	1.54E+01	3.39E+01	5.58E+02	1.32E-02
GAv2	4.92E+00	5.09E-02	1.48E+01	9.25E+01	*1.31E-04*
CCGAv1	6.45E-01	2.18E-01	*3.68E-01*	5.90E-01	2.26E-04
CCGAv2	5.16E-01	*7.90E-02*	2.13E+00	1.26E+00	2.83E-04
DCCGA0	*3.74E-01*	2.22E-01	3.18E+00	5.90E-01	7.13E-04
DCCGA2	4.70E-01	2.05E-01	3.14E+00	7.49E-01	3.23E-04
DCCGA5	6.18E-01	1.97E-01	2.89E+00	2.43E-01	2.83E-04
DCCGA10	1.21E+00	1.74E-01	3.52E+00	*1.34E-01*	3.11E-04
SSGA0	1.98E-01	1.76E+01	2.47E+01	1.27E+02	4.15E-02
SSGA2	1.70E-01	2.11E+01	9.75E+00	1.30E+02	3.99E-02
SSGA5	2.20E-01	2.11E+01	1.02E+01	2.16E+02	6.05E-02
SSGA10	1.72E-01	3.64E+00	3.69E+01	6.21E+02	2.39E-02

Tab. 5. Table displaying the standard deviation of solution results across 20 trials for all 5 functions and 12 experimental groups.

5. Conclusion

Clearly, from the table above, it can be seen that the DCCGA technique was not wholly beneficial in all cases. Indeed, although the DCCGA performs quite well in most of these cases, finding the solution to each function with higher quality than the GA groups and the CCGAv1 group, still it remains obvious that the diffusive techniques were not clearly demonstrated as the reason for increased performance. For the most part, the technique that solved for the higher solution quality in these cases was the CCGAv2 group, the non-diffusive cooperative coevolutionary approach with the higher degree (10%) of elitist preservation imposed. This would seem to suggest that the primary benefit of the algorithm's method was due to its preservation policy, for these functions.

However, having stated that, there is certainly evidence that the diffusion technique *was* beneficial, albeit not the primary contributor to the increased performance. First, functions f_4 and f_5 performed better than the CCGA with 10% elite preservation applied (CCGAv2), although f_5 still did not perform as well as CCGAv1 or GAv2. Secondly, the table above clearly indicates a variation of solution quality depending on the amount of diffusion applied. Moreover, some degree of consistency can be seen in terms of the optimal diffusion rate. In three of the five cases the best diffusion rate was 5%. In the case of f_2, however, it is clear from the above results that the higher degree of elitism was most responsible for the greater solution quality. Further, in four of the five cases the standard deviation was lowest within the DCCGA groups at a rate of diffusion of 5%. In the abhorrent case, function f_2, the best rate was curiously 10%, though the better solution quality occurred with no diffusion (0%). This would seem to suggest that a diffusion rate of 5% would be a good starting point for general problems, but that it may be necessary to tune it for a given function for achieving better results.

There are several possible reasons for the mediocre performance of the DCCGA. First, there exists no real pressure for the model to tune subpopulation size. Future studies of this technique should consider providing a more competitive system for species vying for the same niche. This can be done in many ways, but one such way is to make the diffusion rate increase for a given subpopulation that was not selected to allow the contribution of a collaborator.

Alternatively, diffusion could be presenting added diversity benefits at the start of a run, but impose more disruptive effects as the algorithm converges. A possible solution to this problem might be to reduce the rate of diffusion as the system converges and stabilizes.

Another reason for the poorer than expected performance of this study's model, is the nature of the functions chosen. These functions, chosen for consistency with Potter's studies (Potter et al., 1994) are, for the most part, symmetric along all of their axes. Aside from some ancillary diversity benefit that diffusion in a coevolutionary system might bring to such a function, tuning of the subpopulation sizes may be wholly unnecessary. Future research will consider more asymmetric functions.

Also, there is room for more serious applications of the DCCGA technique in other systems. For instance, in Mitchell Potter's dissertation (Potter, 1997), he explores using a CCGA to build a neural network by constructing the hidden layers, and tuning the initial network. In this model the hidden layers are represented by individual

species which are dynamic in the sense that species are created and become extinct (according to certain rules) as the model finds a solution. A variation of our technique could be used to allow diffusion to be the driving force for the creation and extinction of species. Providing a mechanism to restrict mating of an individual to a group of tags within a specified range, and evaluating by finding components based on discrete groupings of the tags could accomplish this.

The diffusable cooperative coevolutionary model offers many interesting future possibilities for study. Moreover, this study shows that there are more benefits to the DCCGA which remain largely unexplored. We intend to provide further examinations of these issues in the future.

References

1. Cohoon, J. P., Hegde, S. U., Martin, W. N., and Richards, D. "Punctuated equilibria: A parallel genetic algorithm". In: Proceedings of the Second International Conference on Genetic Algorithms. Lawrence Erlbaum Associates (1987) 148-154.
2. Goldberg, D. E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Reading, MA (1989)
3. Goldberg, D. E. and Richardson, J. "Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann, Fairfax, VA (1987) 42-50
4. Fogel, D. B. "Unearthing a Fossil from the History of Evolutionary Computation." To appear in: Fundamenta Informaticae. IOS Press (to appear).
5. Holland, J. Adaptation in Natural and Artificial Systems. The MIT Press, Cambridge, MA (1992)
6. Potter, M. A. and De Jong, K. "A Cooperative Coevolutionary Approach to Function Optimization." In: The Third Parallel Problem Solving from Nature. Springer-Verlag, New York (1994) 249-257
7. Potter, M. A. "The Design and Analysis of a Computational Model of Cooperative Coevolution" (Doctoral dissertation, George Mason University). (1997)
8. Schlierkamp-Voosen, D. and Mühlenbein, H. "Strategy Adaptation by Compteting Subpopulations". In: Parallel Problem Solving form Nature (PPSN III). Springer, Jerusalem (1994) 199-208
9. Spears, W. "Simple subpopulation schemes". In: Proceedings of the Third conference on Evolutionary Programming. World Scientific (1994) 297-307
10. Tanese, R. "Distributed genetic algorithms. In: Proceedings of the Third International Conference on Genetic Algorithms. Morgan Kaufmann (1989) 434-439.