Parameter Control within a Co-operative Co-evolutionary Genetic Algorithm

Antony Iorio and Xiaodong Li

School of Computer Science and Information Technology,
RMIT University, Melbourne VIC 3001, Australia
{iantony, xiaodong}@cs.rmit.edu.au

Abstract. Typically GAs have a number of fixed control parameters which have a significant effect upon the performance of the search. This paper will address the effects of self-adapting control parameters and the adaption of population size within the sub-populations of a co-evolutionary model. We address the need to investigate the potential of these adaptive techniques within a co-evolutionary GA and have proposed a number of model variants implementing adaption. These models were tested on some well known function optimisation problems. The experiments showed that one or more of the model variants investigated yielded improved results over the baseline co-evolutionary model.

1 Introduction

Recently, ecological models of the co-evolution of species have inspired new approaches for GAs [1–3]. Potter and De Jong have proposed a Co-operative Co-evolutionary Genetic Algorithm (CCGA-1) [1] architecture involving multiple co-evolving sub-populations, each dealing with separate parameters of the problem. Sub-populations are evolved much like a typical GA with selection, recombination and mutation operations. The co-evolutionary aspect of this model results from individuals evaluated in terms of individuals from the other sub-populations. This leads to co-operation between sub-populations to attain a mutual goal. By evolving the sub-populations in a modular fashion the problem is also being broken down, assisting the progress of the search, especially with separable problems [1]. This model has been applied to a variety of problem domains with notable success by Potter and De Jong [1, 4].

Another technique which has been found to improve the performance of GAs is the adaption of algorithm control parameters. Control parameters of GAs are usually fixed during a run, whereas self-adaption occurs when control parameters such as mutation or cross-over rate, are represented within the chromosome as a parameter. As a result, they undergo the same evolutionary processes as the problem parameters within the chromosome. Like co-evolution, this is another means of providing a GA with more flexibility to conduct it's search effectively and allows the algorithm to adapt itself to the problem during a run [7, 8]. For more information the reader can refer to Eiben *et al.* [5] where a survey of work in the field of parameter control is provided.

Intuitively the two combined approaches of co-evolution and parameter control should improve the performance of a GA. In this paper we propose a number of model variants of the CCGA-1 which utilise the GAVaPS adaptive population sizing rule originally proposed by Arabas [6]. Bäck *et al.* [9] has also demonstrated the improvements which result from combining the GAVaPS with self-adaption of control parameters.

We study the performance of the adaptive CCGA-1 variants on a number of test function optimisation problems. For the purposes of our comparative study, we use the same test functions that Potter and De Jong used in their original paper [1].

The paper is organized as follows: Section 2 describes the CCGA-1 originally proposed by Potter and De jong [1], and our extensions to the CCGA-1 using self-adaptive genetic operators (e.g., mutation and crossover) and adaptive sub-population size. Section 3 describes the experimental setup including the performance metrics. Section 4 presents the results, and shows that one or more adaptive CCGA-1 variants always performed better than the baseline CCGA-1 upon all of the test functions. The Conclusion section summarizes our observations and discusses directions for future research.

2 The Co-operative co-evolutionary algorithm with adaption

The CCGA-1 model provides a simple model in which it is relatively easy to integrate a variety of evolutionary approaches, while still maintaining the fundamental co-evolutionary behaviour of the algorithm. Consequently, we have chosen the CCGA-1 as a suitable approach to evaluate the performance of self-adaptive crossover and mutation, and the adaption of sub-population sizes according to the Relative Life-time and auxiliary population rules [9].

2.1 The CCGA-1 model

We begin with a description of the fundamental co-evolutionary algorithm employed within our study. The co-operative co-evolutionary genetic algorithm in Figure 1, provides separate populations or species *(s)*, (where the subscript *(s)* represent the sub-population number) for each parameter in the problem domain.

The co-dependent evolution of the parameters is consistent with biological mutualism in that the fitness evaluation of a new individual is done in terms of how much the fittest individuals from the other sub-populations (previous generation) contribute towards its fitness. In other words the algorithm determines how well an individual co-operates with individuals from other sub-populations, the ultimate outcome being a maximisation in co-operation and overall fitness.

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

Fig. 1. The Co-operative co-evolutionary genetic algorithm of De Jong and Potter [1].

2.2 Self-adaptive mutation, crossover, and the selection process

We have applied self-adaption similar to that used by Bäck *et al.* [9]. This requires control parameters of mutation and cross-over to be represented within a chromosome. Each sub-population represents a population of individuals for a particular parameter from the problem domain. The self-adaptive control parameters are encoded in the last two sub-strings of each chromosome, within each sub-population.

The process of self-adaptive mutation we have applied is a two step process; the bits encoding mutation rate at the end of the chromosome are mutated, and the resulting new mutation rate is decoded to be used upon the remaining bits of the chromosome.

Self-adaptive crossover rates are decoded from a sub-string within a chromosome as well. Crossover occurs with a two-point crossover over the entire chromosome after a proportion of the least fit individuals ranked in terms of their fitness, are killed from the sub-population. Two parents are chosen from the remaining sub-population using tournament selection. The probability of crossover for two parents is determined from the average of the two encoded crossover rates. A random number is generated between 0 and 1. If the number is within the range of the average self-adaptive cross-over rate, the parents are mated. The resulting two offspring are mutated with their respective mutation parameters, and reinserted into the existing population. If the crossover test does not succeed the parents are mutated with their respective mutation parameters and reinserted into the sub-population. This process is repeated until the number of individuals which were killed off are replaced. If the number of individuals which are reinserted is odd, one of the replacement individuals is automatically mutated and reinserted into the population.

2.3 Adaptive sub-population size

The *GAVaPS*[6] rule applied to the sub-populations of the CCGA-1, gives individuals a relative life-time (RLT) at the time of creation, which is in proportion to the average fitness of individuals within the population. Bäck *et al.* [9] reconstituted the RLT rules for the problem of function minimisation, which is the variation of the RLT rules we will be applying (equations (1), (2), and (3)).

MinLT and *MaxLT* refer to the static minimum and maximum life-times an individual can have. An individual i with a fitness worse than, or equal to the average can expect a RLT closer to *MinLT*. Likewise, an individual which has a fitness better than the average can expect a RLT closer to the maximum value. *AvgFit* is a measure of the average fitness within the sub-population, and *WorstFit* and *BestFit* are measures of fitness for the least and most fit individuals respectively.

$$RLT(i) = MinLT + \eta . \frac{WorstFit - fitness(i)}{WorstFit - AvgFit} \quad if fitness(i) \geq AvgFit \; . \quad (1)$$

$$RLT(i) = \frac{1}{2}(MinLT + MaxLT) + \eta . \frac{AvgFit - fitness(i)}{AvgFit - BestFit} \quad (2)$$

$$if fitness(i) < AvgFit \; .$$

$$\eta = \frac{1}{2}(MaxLT - MinLT) \; . \quad (3)$$

$$SubPopSize(t+1) = SubPopSize(t) + AuxSubPopSize(t) - D(t) \; . \quad (4)$$

$$AuxSubPopSize(t) = SubPopSize(t) * p \; . \quad (5)$$

Every generation the RLT value of each individual is decremented by 1. When the RLT reaches zero, the individual is removed from the sub-population, unless the individual is the fittest individual in which case it is kept. This is equivalent to an elitist strategy.

Within the approach of Bäck *et al.* [9] eventually all individuals die from old age. Bäck conjectured the high selection pressure resulting from population size minimisation contributed to finding good self-adaptive cross-over and mutation rates. But, this approach can potentially lead to premature convergence with an insufficiently large population. We have decided to counteract this issue by applying the auxiliary replacement equations (4) and (5) proposed by Arabas *et al.* [6]. Combined with a relatively small maximum life-time for individuals we can maintain an appropriate level of selection pressure, while introducing new individuals each generation. At each generation t, we add an auxiliary population *AuxPopSize(t)* and delete *D(t)* of the least fit individuals. The fraction of

new individuals added to the current sub-population is determined by p, where *SubPopSize(t)* is the sub-populations size at that generation. The auxiliary population is added on using the same selection process applied to the replacement of culled individuals.

3 Experimental design

The CCGA-1 was evaluated with a number of adaptive variations as shown in Table 1 using the parameters of Table 2. We employ a cap of 500 individuals for the adaptive sub-population size to make sure the population does not increase to an unnecessarily large number. The models we investigated were tested on the same set of functions used by Potter and De Jong [1] (see Table 3).

Table 1. Descriptions of the algorithm designations.

Algorithm designation	description
CCGA-1	Baseline co-evolutionary model
CCGAM	CCGA-1 with self-adaptive mutation rate
CCGAMX	CCGAM with self-adaptive crossover rate
CCGAAP	CCGA with adaptive sub-population sizing
CCGAMAP	CCGAAP with self-adaptive mutation rate
CCGAMXAP	CCGAAP with self-adaptive crossover rate

For each run of a particular algorithm, we terminated after 200,000 evaluations. For each function we conducted 30 runs and acquired the best mean fitness and standard deviation for each of the 200,000 evaluations over those 30 runs. The evaluation count is used as a time metric for establishing a comparative baseline performance between the models. This is done for two reasons, firstly the CCGA-1 model has a significantly larger number of evaluations per generation although its overall performance is typically better than a single population GA. Secondly, the adaption of population size within a number of the models results in a varying number of evaluations per generation.

4 Results

Table 4 presents the best mean fitness after 200,000 evaluations for each of the models upon each test function. A number of comparative plots of each algorithm's performance are provided for each test function investigated in Figure 2. The performance of each algorithm is ranked in terms of the best results found for

Table 2. Algorithm parameters and features.

Parameter	Value
chromosome representation	16 bits for a function variable, and 8 bits for each self-adaptive parameter.
chromosome length	32 bits
selection	rank selection and tournament selection
genetic operators	two-point crossover with bit-flip mutation
static crossover probability	0.6
static mutation probability	1/(chromosome length)
self-adaptive mutation probability range	0.001 to 0.25
self-adaptive crossover probability range	0.6 to 1.0
kill percentage	10%
sub-population size	100 (initial size which is fixed for all CCGA-1 model variants which do not incorporate adaptive sizing)
maximum sub-population size	500 (the maximum size of a sub-population within the CCGA-1 models incorporating adaptive sizing)
sub-population initialisation	random
MinLT	1
MaxLT	4
p	0.7

Table 3. Test functions utilised to evaluate the performance of the algorithms. n is the dimension of the function, and R is the range of the function variables.

Name	Function	n	R
Rastrigin	$f_1(\overrightarrow{x}) = 10.0n + \sum_{i=1}^{n}(x_i^2 - 10.0\cos(2\pi x_i))$	20	[-5.12,5.12]
Schwefel	$f_2(\overrightarrow{x}) = 418.9829n - \sum_{i=1}^{n} x_i \sin(\sqrt{\lvert x_i \rvert})$	10	[-500,500]
Griewangk	$f_3(\overrightarrow{x}) = 1 + \sum_{i=1}^{n} \frac{x_i^2}{4000} - \prod_{i=1}^{n} \cos(\frac{x_i}{\sqrt{i}})$	10	[-600,600]
Ackley	$f_4(\overrightarrow{x}) = 20 + e - 20.\, e^{(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2})}$ $-e^{(\frac{1}{n}\sum_{i=1}^{n}(\cos 2\pi x_i))}$	30	[-30,30]
Rosenbrock	$f_5(\overrightarrow{x}) = 100(x_1^2 - x_2)^2 + (1 - x_1)^2$	2	[2.048,2.048]

Table 4. Best fitness and standard deviation after 200,000 evaluations.

Algorithm	Rastrigin	Schwefel	Griewangk	Ackley	Rosenbrock
CCGA-1	0.1540 ±0.3687	0.08393 ±0.09581	0.04734 ±0.02994	0.01227 ±.005645	0.02046 ±0.03322
CCGAM	1.921 ±1.317	1.874 ±6.833	0.3494 ±0.1364	1.247 ±0.5214	0.005109 ±0.01043
CCGAMX	2.708 ±1.425	1.045 ±4.594	0.3017 ±0.1431	0.9333 ±0.5245	0.004421 ±0.005681
CCGAP	0.001311 ±0.005653	0.0006013 ±0.0008367	0.03431 ±0.03394	0.005765 ±0.001606	0.03454 ±0.07332
CCGAMAP	2.078 ±4.411	0.02420 ±0.05113	0.06615 ±0.02763	7.646 ±2.579	0.002995 ±0.005983
CCGAMXAP	0.9237 ±1.914	0.01351 ±0.03134	0.05635 ±0.02054	6.359 ±3.228	0.002943 ±0.006692

a particular algorithm variation and test function within Table 5[1]. The ranked value is between 0 and 5, where a ranking of 5 represents the best performance of a particular algorithm and test function. The ranked values are summed across test functions, for each algorithm. The sum provides an indication of the overall ranking of the algorithms investigated.

Table 5. Comparative ranking of the algorithms upon the test functions, in terms of the best individual found after 200,000 evaluations.

Test function	CCGA-1	CCGAM	CCGAMX	CCGAAP	CCGAMAP	CCGAMXAP
Rastrigin	2	4	6	1	5	3
Schwefel	4	6	5	1	3	2
Griewangk	2	6	5	1	4	3
Ackley	2	4	3	1	6	5
Rosenbrock	4	3	2	5	1.5	1.5
Total	14	23	21	9	19.5	14.5
End Ranking	2	5	4	1	3	2

We observe from Figure 2 that one or more of the adaptive CCGA-1 models evaluated, performed better than the baseline CCGA-1 upon all of the test functions.

[1] The CCGAMAP and CCGAMXAP are so close in their best individual found for the Rosenbrock function, that we have split the ranking evenly between them.

Bäck *et al.* has hypothesized that the main source of the improvement in his experimentation of adaptive and self-adaptive GAs was the adaption of the population size. This is consistent with our observations as well, in that the CCGAAP outperformed the CCGA-1 upon the Schwefel, Rastrigin, Griewangk, and Ackley functions. From Table 5 we see that the CCGAMXAP also performed as well as the CCGA-1 baseline. It is apparent that the CCGAMXAP performed significantly better upon the Rosenbrock and Schwefel functions, where the CCGA-1 performed more poorly.

5 Conclusion

We have established that one or more of the adaptive models performed better than the baseline CCGA-1 on all of the test functions we investigated.

To summarise, we have made the following observations:

- The CCGAAP model performed as well or better on all functions except the Rosenbrock function. This suggests that some form of adaptive population sizing can generally yield improvements to the CCGA-1. The application of the *RLT* rule assists in removal of individuals from within sub-populations which are deemed to make a poor contribution towards the fitness evaluation. At the same time, the lifespan of these individuals is sufficient for any beneficial future contributions to be made.
- All the models incorporating self-adaption obviously involve some degree of learning to establish appropriate mutation and/or crossover rates as the algorithm progresses. This learning is part of the search process, and it is undertaken by each individual in each sub-population. We can see from our experimental results that self-adaptive CCGA-1 models generally converges quite slowly and also prematurely converge. We hypothesize that the poor performance of these models is the result of the large number of self-adaptive parameters involved in the search. Within these models we are not just searching for good solutions, but also good control parameters. As a result the search space is significantly larger than it would be otherwise. Within the self-adaptive CCGA-1 models we are searching a much larger number of parameter values, which suggests significant time is taken away from the search for good solutions.
- We have observed improvements with both the CCGAMAP and CCGAMXAP over the baseline CCGA-1 upon the Schwefel and Rosenbrock function. Compared with the approach of Bäck *et al* [9] where the population diminishes over successive generations, the sub-populations in our approach do not diminish, but increase in size towards a cap of 500 individuals. It is reasonable to conclude in our case that the selection pressure resulting from the comparatively smaller maximum life-time for individuals helps to remove individuals with unsuitable self-adaptive mutation and crossover values.
- Because of the nature of the CCGA-1 model which separates the object variables into sub-populations, we mutate each of the object variables independently within a self-adaptive scheme. This results in a greater exploration

of the search space. It also suggests that good results upon uniform fitness landscapes such as the Rosenbrock function can be expected, where a high degree of exploration is desirable to find a good direction in the search.
- The RLT rule, with a short maximum lifetime of 4 introduces significant selection pressure to the search, by allowing individuals which have not contributed after 4 generations to die off and be replaced. This is the element of the CCGAMXAP and CCGAMAP model which provides a force for the survival of relatively fitter individuals within sub-populations.

Overall we see that adaptively sizing the sub-populations using the *RLT* rule yielded the largest improvements, and further research in the area of adaptively sizing sub-populations would be desirable. The highly adaptive nature of the models we investigated suggests that they may be suitable for optimisation problems involving non-stationary environments [10].

References

1. Potter, M. A., De Jong, K. A.: A Cooperative Co-evolutionary Approach to Function Optimization. In: Davidor, Y., Schwefel, H.-P., Manner R. (eds.): Parallel Problem Solving from Nature PPSN-III., Proceedings of the International Conference on Evolutionary Computation, Lecture Notes in Computer Science Vol. 866. 249–257 (1994)
2. Husbands, P. and Mill, F.: Simulated Co-Evolution as The Mechanism for Emergent Planning and Scheduling. In: Belew, R. and Booker, L. (eds.): Proceedings of The Fourth International Conference on Genetic Algorithms 264–270 (1991)
3. Hillis, W. D.: Co-evolving Parasites Improve Simulated Evolution as an Optimization Procedure. In: Langton, C. G., Taylor, C., Farmer, J. D., and Rasmussen, S. (eds.): Artificial Life II, Santa Fe Institute Studies in the Sciences of Complexity Vol. X 313–324 (1991)
4. Potter, M. A., and De Jong, K. A.: Cooperative Co-evolution: An Architecture for Evolving Coadapted Subcomponents. In: Evolutionary Computation, Vol. 8., No.1, 1–29, (2000)
5. Eiben, A., Hinterding, R. and Michalewicz, Z. Parameter Control in Evolutionary Algorithms. In: IEEE Transactions on Evolutionary Computation Vol. 3 No. 2 124–141 (1999)
6. Arabas, J., Michalewicz Z., and Mulawka, J.: GAVaPS - A Genetic Algorithm with Varying Population Size. In: IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on Evolutionary Computation Vol. 1. 73–78 (1994)
7. Bäck, T. and Schutz, M.: Intelligent mutation rate control in canonical genetic algorithms. In: Ras, Z. W. and Michalevicz, M. (eds.): Foundations of Intelligent Systems, Ninth International Symposium ISMIS'96, Lecture Notes in Artificial Intelligence, Vol. 1079. 158–167 (1996)
8. Hinterding, R.: Gaussian mutation and self-adaptation for numeric genetic algorithms. In: Proceedings of 1995 IEEE International Conference on Evolutionary Computation 384–389 (1995)
9. Bäck, Th., Eiben, A. E. and van der Vaart, N. A. L.: An empirical study on GAs without parameters. In: Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E.,

Merelo, J. J., and Schwefel, H.-P.(eds.): Parallel Problem Solving from Nature — PPSN V, Lecture Notes in Computer Science Vol. 1917 315–324 (2000)
10. Grefenstette, J.J.: Genetic Algorithms for changing environments. In: Manner, R. and Manderick, B. (eds.): Parallel Problem Solving from Nature — PPSN II. Elsevier Science Publishers B.V., 137-144 (1992)

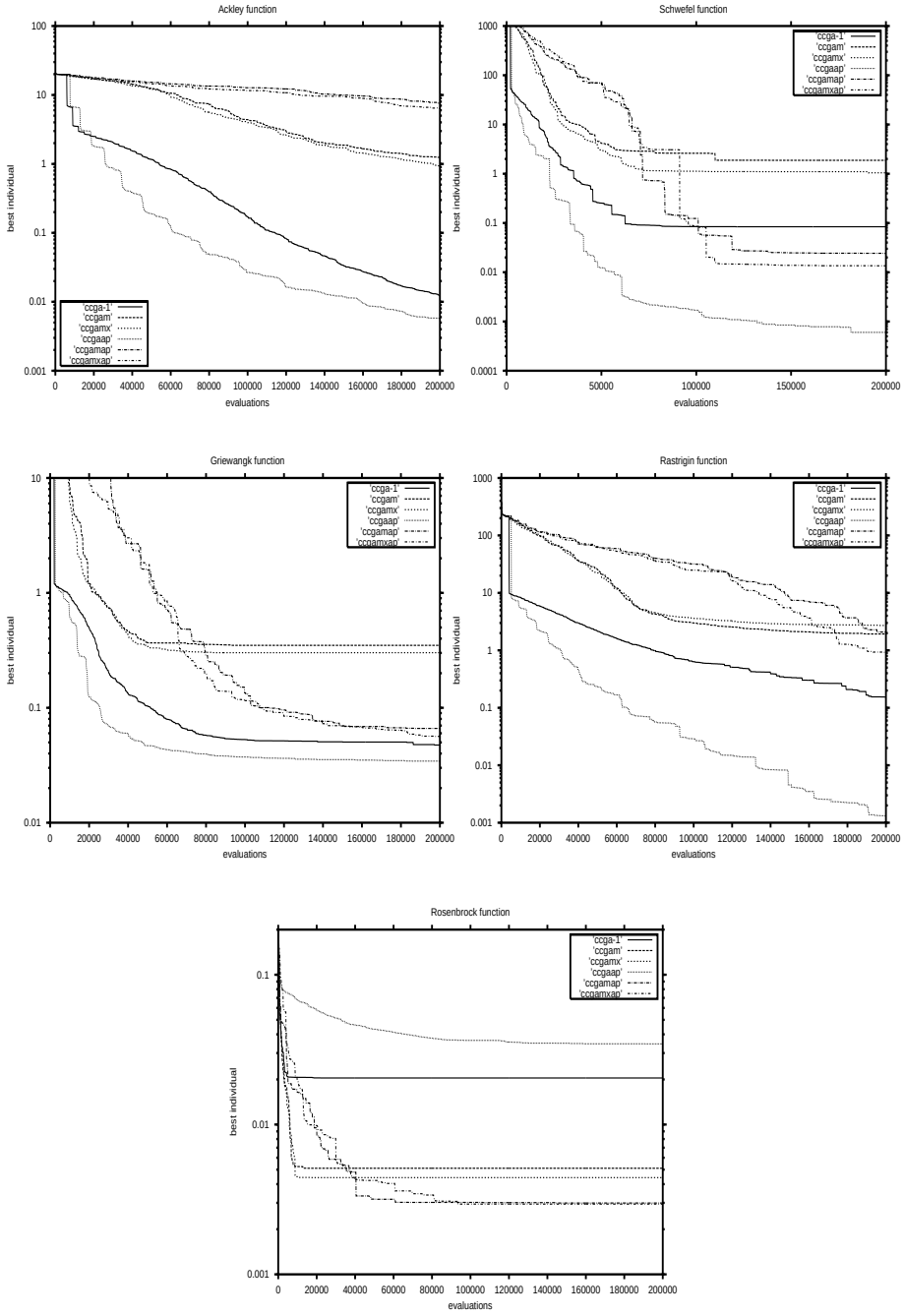

Fig. 2. Best fitness performance of the adaptive CCGA-1 algorithms upon the Ackley, Schwefel, Griewangk, Rastrigin, and Rosenbrock functions.